Algorithmen und datenstrukturen in c pdf

In diesem Kapitel werden Themenstrukturen, Zeiger und dynamische Speicherverwaltung gemischt. Was auf den ersten Blick ein wenig kompliziert aussieht - und manchmal auch das - ist eine große Erleichterung, wenn man es beherrscht. 21.1 Zeilenlisten (nur Kettenlisten) In Kapitel 14, Dynamic Memory Management, habe ich ausführlicher erklärt, wie man mit dynamischem Speicher umgeht. Dynamisch bedeutet, dass Speicher während des Programms einem Haufen ähnelt. Der Hauptzweck dynamischer Datenstrukturen besteht darin, dass die Struktur durch einen Zeiger des Strukturtyps selbst definiert wird. Beginnen wir mit der folgenden Mitarbeiterlistenstruktur: Struct-Datum; int-Tag; int Monat; int Jahr; ; struct verwendet den Namen char 20;char Vorname; Ändern der Struktur des Datums eingest Datumstrukturen; lange senk; Eine solche Struktur wurde bereits erwogen und ist daher nicht neu. Nun muss diese Struktur erweitert werden: struct datum int tag; int Monat; int Jahr; ; struct employed - char name; char Name; struct datum ändern; struct datum eingest; lange senk; struct employed (nächste; nächste Zeile kann Ihnen am Ende der Struktur auftreten: Das Besondere an diesem Zeiger ist, dass es sich um einen Zeiger auf die Adresse handelt, die den gleichen Typ wie die Struktur selbst enthält (die Struktur wird verwendet). wobei das nächste Element mit einem Zeiger anstelle einer Indizierungsanweisung angezeigt wird und Sie den Speicherplatz für das nächste Element im Voraus reservieren sollten. Darüber hinaus braucht die Kette ein Ende. Verwenden Sie dazu einfach den folgenden Zeiger und geben Sie ihm einen Nullzeiger: die Strukturen, die vom nächsten NULL verwendet werden; So wird die Struktur verwendet aussehen, wie in Abbildung 21.1 gezeigt. Klicken Sie hier, um das Bild von Abbildung 21.1 Struktur für die Single-Chain-Liste erneut zu vergrößern: Die als nächstes verwendete Zeigerstruktur gibt sich nicht selbst an, sondern adressiert das nächste Element desselben Typs. (Erinnerung: Dereferenzieren Zeiger Adresse, nicht Wert.) Dieses Beispiel bezieht sich zunächst auf NULL, da die Daten noch nicht eingegeben wurden. In Kapitel 15, Strukturen, habe ich bereits gezeigt, wie man auf einzelne Elemente der Struktur zugreifen kann, zum Beispiel: dann a.name, a.vorname oder Alter, etc. Zugriff auf einzelne Strukturelemente. Ähnliche dies, wenn Sie nicht mit einer variablen Struktur arbeiten, sondern mit Zeigern auf die Struktur: die Struktur wird verwendet structzeiger; Der Zugang zu bestimmten Elementen der Struktur ist, als ob es sich um einen strukturellen Seiger) Namen (structureseiger) handelt. Glücklicherweise haben Compiler-Builder einen zusätzlichen Operator erstellt, der eine Kombination aus Dereferenzierungs- und Zugriffselement ist: Da der Operator die Form eines Zeigers hat, ist er auch leichter zu lesen. Dies führt zu folgendem Schreiben, um auf jedes Element der Struktur zuzugreifen: Aber irgendwann sollten Sie das Dataset beenden oder erneut sortieren. Daher muss die Kette gestartet werden, d. h. die ursprüngliche Adresse, auf der die Liste ausgeführt wird. Daher ist ein weiterer Zeiger auf die Struktur erforderlich, in der sich die ursprüngliche Adresse befindet: die Am Anfang aufgenommene Struktur; Da der Datensatz nicht am Anfang des Programms eingegeben wurde, bezieht sich dieser Zeiger ursprünglich auch auf NULL. Bisher sieht die Struktur wie folgt aus: struct datum int tag; int Monat; int Jahr; struct employed - char name; char Name; struct datum ändern; struct datum eingest; lange senk; struct employed (next; structures used next and NULL; structures used by start and NULL; Now there is #include #include #define #include linear_list1 a feature that you to attach sets of addresses ateach each other. structemployed .char name (MAX); char firstname-MAX; struct date alter; struct date eingest; long gehalt; struct/'Wir fügen Datensatz an oder geben neu ein: n'name,v'vornam,at'alter.tage,am'alter.month,aj'alter.jahr yr g'content s/ void anhaengen (char n char q v, int at, int am, int aj, int eint, int einm, int einj, long g) verwendet Zeiger; Wir fragen, ob es bereits ein Element in der Liste. Wir suchen nach einem Punkt, auf den unser Zeiger und Ausgangspunkte. Wenn Start noch zeigt auf NULL, dann beginnen Sie immer die Adresse unseres ersten Elements und - die Überschrift (oben) unserer Liste. Die Frage/wenn (anfanf - NULL) - / Wir reservieren Platz für unsere Struktur, um das erste Element der Liste zu sein.Zunächst einmal); Rückgabe strcpy (Anfangsname, n); strcpy (Anfangsname, v); start--zgt.alter.tag zum Preis; Anfang des Monats. Anfang des Jahres und aj; start--'gt;gest.tag und eint; Anfang des Monats. Jahr - ein Jahr; Inhalt und g; Wir haben also ganz oben auf der Liste. Von nun an zeigt der Zeiger immer auf das Element davor. Da dies nun das erste Element in der Liste war, beginnt der Zeiger auf den nächsten Zeiger zu zeigen. Die nächste zeigt Null am Ende und wieder. Das B/Start-up ist unynementedNULL; Es scheint bereits mindestens ein Element in der q-Liste zu sein, da der Anfang nicht NO NULL ist. Jetzt suchen wir nach dem nächsten Element, bis der nächste Zeiger NULL verletzt. Wir haben also das Ende der Liste gefunden und können einen neuen Datensatz anfügen. In/ noch ein Indikator anfang; Wir zeigen auf eines der Zgt; qgt-Elemente. Rückgabe Zeigerzeiger-z-gt;sn vor; / - Zeiger auf den neuen Speicherplatz (Strcpy (Zeigername, n); strcpy (pointer, name, v); pointer-zgt.alter.tag'at; pointer-zgt.age.month.am; pointer-zgt.year'aj; pointer zgt;eingest.tag'eint; pointer zgt;eingest.month.one; pointer-zgt.jahr'einj; We're shifting the data structure. int atag,amon,ajahr,eintag,einmon,einjahr, einjahr, einjahr, einjahr; long gehalt; printf; printf (Name.....; fgets (us, MAX, stdin); printf (Vorname.....:..... fgets (vorne, MAX, stdin); printf (Alter..... (tt.mm.jjjj):); scanf (%2d.%2d.%4d, zatag, Amon, sion); printf (Set on.. (tt.mm.jjjj):); scanf (%2d.%2d.%4d, eiintag, einmon, ein Jahr); printf (Monatsgehalt.....:::::); scanf (%ld, Inhalt); getchar (); /- Anhängung des Datensatzes, von hinten eingeführt - anhaengen (uns, vorne, atag, amon, ajahr, aintag, Einmon, ein Jahr, Gehalt); - int main (void) - während (1) eingeben (); EXIT_SUCCESS; Zuerst wird die Eingabefunktion aufgerufen, um einzelne Daten einzugeben. Diese eingeführten Variablen werden dann als Argument an die anhaengen Funktionsparameter weitergegeben. Die Linie ist sehr wichtig für die Funktion. Ein Element soll bereits eingegeben worden sein, daher ist der nächste Punkt das zweite Element in der Liste. Wenn die Zeiger nicht stattdessen auf die Zeigeradresse zeigen, handelt es sich nicht um einen syntaktischen Fehler, aber es wird immer die erste Struktur neu geschrieben. Die nächste Zeile ermöglicht es, während der Zyklus zuerst funktioniert Geben Sie die Daten ein: Andernfalls, während der Zyklus mit der Abfrage fragt, ob der Zeiger den nächsten angibt, nie richtig abgebrochen wird, weil er nie auf NULL verweist: während (zeiger-zgt'next !) zeiger-zstep; Das muss nun veranschaulicht werden. Zwei Personen wurden bereits eingeführt. Infolgedessen gibt es zwei Datensätze (siehe Abbildung 21.2). Klicken Sie hier, um Abbildung 21.2 der Kettenliste der beiden Einträge hier zu vergrößern. Der nächste Strukturzeiger im letzten Element zeigt immer auf NULL und zeigt daher immer auf das Ende der Kette. Als Nächstes möchten Sie ein Feature erstellen, mit dem Sie einzelne Elemente in der Liste entfernen können. Space wird wie gewohnt freigegeben, mit dem Feature frei ().21.1.1 Löschen Sie das erste Element in der Liste Wenn Sie das erste Element in der Liste entfernen möchten, ist es nicht allzu schwierig. Nur ein Strukturtypzeiger sollte die Adresse von start--'gt;next (zweites Element) angeben. Dann kann der Speicher mit einem freien (Anfang) freigegeben werden. Schließlich beginnt der Zeiger mit der Adresse des Zeigers. In der Praxis werden Sie das erste Element auf der Liste los: Die q/ void loesche (char s wen) s struct (pointer; qgt; qgt;) gibt es überhaupt ein Element? Derzeit ist das erste Element in der Liste ein Element, das nach Entfernung gesucht wird. So ergibt sich für den Moment der folgende Zustand: Klicken Sie hier, um das Bild von Abbildung 21.3 Zeiger auf das folgende Element von Anfang an zu zoomen Nun folgt der Aufruf: Hier wird der Abstand frei, auf den der Zeiger beginnt: Klicken Sie hier, um Abbildung 21.4 zu vergrößern. Wenn Sie es jetzt darauf belassen, geht der Rest der Kettendaten wahrscheinlich verloren, da es keinen Start mehr gibt. Die Zeigeradresse sollte oben auf dem Zeiger angegeben werden: Dies führt zu folgendem endgültigen Bild: Klicken Sie hier, um das Bild zu vergrößern Abbildung 21.5 Der Index des ersten Elements erhält eine neue Adresse.21.1.2 Entfernen eines Elements in der Liste Erstes Element in der Liste war nicht schwer zu entfernen. Anders verhält es sich, wenn das Element irgendwo in der Liste entfernt werden muss. Dies erfordert einen weiteren Zeiger: den Zeiger, der auf die Adresse zeigt, die sich vor dem Element befindet, das entfernt werden muss, und einen weiteren Zeiger, der auf das nächste Element zeigt, nachdem das Element entfernt wurde. Hier ist eine kurze Zusammenfassung der vorherigen loesche (): /Funktion zum Entfernen von th/ void loesche (char Wen) - Strukturen besetztzeger, zeiger1; / Gibt es überhaupt ein Element? Die Frage/ ob (start! - NULL) - ist unser erstes Element, das wir suchen (wer)) gibt es? Die Frage/if (strcmp (beginning-'gt;name, wen) - 0) zeiger-start-'gt;next; Kostenlos (Start); start'zeiger; .else . . . . . Wir schauen in die weitere Kette, um zu sehen, ob der zu entfernende Artikel vorhanden ist - zeiger'start; Dies führt derzeit zu folgender Grundposition: Klicken Sie hier, um das Bild von Abbildung 21.6 zu vergrößern Bei der Suche nach einem zu entfernenden Element, vorausgesetzt, dass das gesuchte Element das zweite Element in der Liste ist (siehe Abbildung 21.6). Ein Teil des Quellcodes, der nach einem bestimmten Namen in der Liste sucht, sieht wie folgt aus: während (zeiger-zgt'next! - NULL) - zeiger1'pointer-gt;next; / Ist die zeiger1 Adresse der Name, den Sie suchen? If (srkmp (senger1-gt; Name, wen) - 0. . . . . Bevor Sie jetzt Platz machen können, bezieht sich das Element in der Zeigerliste auf die Adresse für den nächsten Zeiger, sodass die Kette nicht bricht: der Zeiger - 'next'pointer1-'gt'next; Kostenlos (zeiger1); Eine Pause; Zeiger1; Das Ende!!! As a . . . . . Ich habe schon ---------- Schauen wir uns das genauer an: zeiger-'zeiger1-'gt'next; In Worten: Der Zeiger, der auf die Adresse der nächsten (nächsten) Datenstruktur zeigt (im Moment gibt der Zeiger- qgt;disalceive noch das Element an, das entfernt werden muss, auf das der Zeiger1 zeigt), erhält nun die Adresse, an die der nächste Elementzeiger entfernt wird (zeiger1) Punkte. Es ist einfacher, aus dem Diagramm zu verstehen: Klicken Sie hier, um Abbildung 21.8 zu vergrößern. Daher wird das Element, das entfernt werden muss, nicht montiert. Jetzt kann der Speicherplatz freigegeben werden: Dies führt zu folgendem Bild: Klicken Sie hier, um das Bild bild 21.9 zu vergrößern. Übertragen Sie zunächst den Zeiger an die oberste Adresse der Liste, und übergeben Sie die Liste, bis der Zeiger den NULL-Punkt angibt, der das Ende der Liste darstellt. Hier sehen Sie die volle Funktion für die Kettenliste: ungültige Ausgabe (Leerheit) - Strukturierung, die vom Seeger und dem Anfang verwendet wird; Der Anfang ==========||); printf (%10cImim%10c Geburtsdatum endet Gehalt, ','); printf (Ich bin a) während (Zeiger! - NULL) sprintf (%12s,%-12s%0 2d.%0 2d.%0 4d %02d.%0 2d.%0 4d-%0 6ld, zeiger- name, pointer, pointer- zgt'age.tag, pointer-zgt.month, pointer--gt;year, pointer--gt;-gt;content printf (---------- ----------) ; Zeigerzeiger-z-gt;sn vor; Und jetzt sehen Sie das gesamte Programm, einschließlich der Hauptfunktion (): / linear_list2.c. s/ #include <stdio.h>#include #include #define MAX <string.h> <stdlib.h>20 struct datum int tag; int Monat; int jahr; s; structemployed char vorname (MAX); Ändern des Datums der Struktur Strukturierung des Eingest-Datums; lange senk; Struct verwendet next; s; Struct, die von next NULL verwendet wird; struct, verwendet start NULL; / Wir fügen einen Datensatz an oder geben einen neuen ein: n'name,v'vornam,at'alter.tage,am'alter.monat.a. Year g'content q/ emptiness anhaengen (char zen, char q v, int at, int am, int aj, int eint, int eint, int einj, long g) . . . . . He said , he said that he and I . . . . . Wir suchen nach einem Punkt, auf den unser Zeiger und Ausgangspunkte. Wenn Start noch zeigt auf NULL, dann beginnen Sie immer die Adresse unseres ersten Elements und - die Überschrift (oben) unserer Liste. Frage/wenn (Start - NULL) - /- Wir lassen Raum für unsere Struktur für das erste Element in der Liste. Rückgabe strcpy (Anfangsname, n); strcpy (Anfangsname, v); start--zgt.alter.tag zum Preis; Anfang des Monats. Anfang des Jahres und aj; start--'gt;gest.tag und eint; Anfang des Monats. Jahr - ein Jahr; Inhalt und g; Wir haben also ganz oben auf der Liste. Von nun an zeigt der Zeiger immer auf das Element davor. Da dies nun das erste Element in der Liste war, beginnt der Zeiger auf den nächsten Zeiger zu zeigen. die nächsten Punkte auf NULL am Ende und wieder. Das B/Start-up ist unynementedNULL; Es scheint bereits mindestens ein Element in der q-Liste zu sein, da der Anfang nicht NO NULL ist. Jetzt suchen wir nach dem nächsten Element, bis der nächste Zeiger NULL verletzt. Wir haben also das Ende der Liste gefunden und können einen neuen Datensatz anfügen. Die Frage/ noch beginnt der Index; Wir weisen auf ein 1. Element hin. ()) q NULL) fprintf (stderr,' возвращение; • . . . . . / - . . . . . n); strcpy (-> > ,v); . . . . . >eingest.tag'eint; .eingest.month.one; . . . . . •-€€€ content'g; »->'-NULL; • /* Funktion zum Löschen einer Datei */ void loesche(char *wen) - struct employed *zeiger, *zeiger1; /* Gibt es überhaupt ein Element? */ if(beginning != NULL) - /* Ist unser erstes Element das, was wir suchen (wer[])) ist? */ if(strcmp(start->name,wen) == 0) zeiger=start->next; free(Anfang); start=zeiger; else x /* Es ist nicht das erste zu löschende Element. * Wir suchen in der weiteren Kette, um zu sehen, ob das zu löschende Element vorhanden ist. */ zeiger=anfang; while(zeiger->next != NULL >) */ if(strcmp(zeiger1->name,wen) == 0) . > > . . . . . else */ x /* End if(beginning != NULL) */ else printf(Es gibt keine Daten zum Namen!!!); ========== ==========||); printf (%10c'%10c' ' ' ' ' ' , ' , ' , ' ' , ' . . . . . printf (== ==========|||; Sprintf (%12s,%-12s %0) 2d.%0 2d.%0 4d %02d.%0 2d.%0 4d-%0 6ld, ... printf (---------- ----------) ; • . . . . . - char nam'MAX,front-MAX; int atag,amon,ajahr,eintag,einmon,einjahr; lange senk; char 'ptr; printf (B.....:..... fgets (, MAX, stdin); ptr strrchr (, "); PTR – 0; printf (b.....::); fgets (, MAX, stdin); ptr strrchr (, "); PTR – 0; printf (..... (tt.mm.jjjj):); scanf (%2d.%2d.%4d, ... printf (. (tt.mm.jjjj):); scanf (%2d.%2d.%4d, eintag, einmon, ... printf ()... .....::: scanf (%ld, b); getchar (); anhaengen (, vorn, atag, amon, ajahr, eintag, einmon, , , ,

```
/* ifcende == NULL) - if((ende= malloc(sizeof(struct employed))) == null) - printf(Könnte keinen Speicherplatz für Endreserve haben); •
(EXIT_FAILURE); Mal ist bereits ein Element in der Liste vorhanden. Wir suchen nach einem Artikel, bei dem sich unser Startpunkt befindet. Wenn Start noch auf NULL zeigt, dann enthält start die Adresse unseres 1. Absatzes und damit die Übeschrift (der Anfang) unserer Liste. Die Fragel ob (Anfang - NULL) s /) Wir reservieren einen Platz für unsere Struktur - für das erste Element der Liste. Rückgabe Zähler strcpy (Anfangsname, strtok (n, ) ) ; strcpy (Anfangsname, strtok (v,) ) ; ); anfang &gt;age.month.am; anfang &gt;age.year'aj; start-&gt;engest.tag'eint; anfang &gt;month.one; start-&gt;engest.jahr'eing; &gt;
content'g; /Hier, y nec ecth neue nenzig ecnicte. C otoro bremen yrzeztern zu yrzeztern - for das erste Element der Liste. Rückgabe &gt;age.year'aj; start-&gt;engest.tag'eint; anfang &gt;month.one; start-&gt;engest.jahr'eing; &gt;
content'go; Der Starter ist ein spitzer Zeiger; Der Starter erwartet NULL; Die letzte Option ist, dass wir ein Element irgendwo dazwischen einfügen müssen. Die Frage/sonst; ob (compare (zeeger, zeiger2))Wir suchen nach einem Element, das vor dem Zeiger und Zeiger steht. (zeiger1-'gt;next !
zeiger1'zeiger1-'gt'next; zeiger-malloc (Größe (Struktur nun mit einem zusätzlichen Zeiger auf den Vorgänger aus (siehe Abbildung 21.22). Klicken Sie hier, um das Bild von Abbildung 21.22 der Doppelstruktur zu vergrößern Liste Bevor all dies in die Praxis umgesetzt wird, bekommen Sie
schnell eine Vorstellung davon, wie man sich eine zweigleisige Kettenliste vorstellt (siehe Abbildung 21.23). Klicken Sie hier, um das Bild von Abbildung 21.23 Double Chain Liste kommende Seite Neuschreibung Features im Einzelnen Kettenlisten Abschnitt verwendet zu vergrößern, so dass sie mit
doppelKette Listen verwendet werden können. Sie sollten immer sicherstellen, dass jedes Element in der Liste jetzt einen Vorgänger hat. Beginnen wir mit der Funktion von anhaengen (): Leere von anhaengen (char zen, char y v, int at, int am, int aj, int eint, int eing, long g) s /) ein Zeiger, um
auf einzelne Elemente zuzugreifen - Struktur en, die am Zeiger, zeiger1; / Bereits ein Speicher für den letzten Zeiger zur Verfügung gestellt? Die Frage/ ob (ende - NULL) - wenn ((ende'malloc (sizeof (struct employed) ) - null) - printf (Kann keinen Platz für die Lagerung für den Endlagerbestand haben);
Rückgabe wir fragen, ob es bereits einen Punkt auf der Liste gibt. Wir suchen nach einem Element, auf dem unser Start-Zeiger hinweist. Wenn Start noch auf NULL zeigt, dann ruft start die Adresse unseres ersten Elements ab und ist daher der Titel (Start) unserer Liste. If (angang) ...
zeiger'start-'gt;next; Wenn (zeiger - NULL) - frei (start); start'null; end'null; &gt; Rückgabe &gt;
```

```
Element in der Liste, und das erste Element, das Sie eingeben, ist das erste Element. 2. Das eingefügte Element ist das kleinste und wird am Anfang eingefügt. Letztere Option ist auch die schwierigste Option. Das Element muss irgendwo
in der Mitte eingefügt werden. Das nächste Feature überprüft, welche Option wahr ist, und führt dann den Auftrag aus. Zuerst; Funktionsmanager: void sorter_eingeben (char n, char th v, int am, int aj, int, int em, int ej, long geh) Jetzt ist es notwendig zu prüfen, ob es ein Thema in der Liste gibt: wenn (anfan
und NULL) anhaengen (n,v,at,am, aj,et,em,ejh); Wenn Sie noch kein Element in der Liste haben, wird anhaengen mit entsprechenden Argumenten aufgerufen. Es ist bereits mindestens ein Element in der Liste vorhanden. Die Suche danach beginnt also mit: zeiger'start; während ('Zeiger!' NULL (strcmp (
zeiger- > pointer-pointer-zgt; Einzelne Elemente in der Liste schneiden sich, bis wieder das neue Element größer oder gleich dem Namen ist, auf den sich der Zeiger bezieht. So oder so wird der Zyklus unterbrochen. Jetzt müssen Sie überprüfen,
warum der Zyklus unterbrochen wurde. Zuerst können Sie sehen, ob es eine Übereinstimmung gab, und das neue Element wird von hinten kompensiert: if (seiger und NULL) anhaengen (n,v,at,am,aj,et,em,aj,geh); In diesem Fall ist das neue Element das größte und nähter an der anhaengen Funktion am
Ende. Nächste Option: Das neue Element ist das kleinste und sollte an der Spitze der Liste platziert werden: wenn (zeiger - der Anfang) anfang'malloc (größe (Struct employed)); strcpy (start->name,v); strcpy (start->vorname,v); start->start->age.month'am; start->age.year'aj; start-
>engest.tag'et; start->engest.monat'em; start->engest.jahr'ej; start->content-go; start->next'pointer; Es sieht so aus: mehr, wenn (Zeiger) hier klicken, wird das Bild von Abbildung 21.14 zu vergrößern. start'malloc (Größe (Strukturierung wird verwendet)) Klicken Sie hier, um das Bild von Abbildung 21.15
Speicherplatz zu vergrößern, der für das neue Element wird am Anfang eingefügt. Jetzt fehlt die schwierigste Option: Das Element muss irgendwo in der Mitte eingefügt werden: sonst zeigler1 es Anfang; Wir
suchen einen Artikel, der vor dem Zeiger steht. In/ while (zeiger1-'gt;next ) zeiger1'pointer1-'gt'next; zeiger-malloc (Größe (Struktuierung)); strcpy (Name,n); strcpy (Name,v) index-alter.tag'at; index-zgt.age.month.am; pointer-zgt.year'aj; Zeiger-'gt;engest.tag'et; Zeiger qgt'eingest.month'em;
Zeiger auf das 'Jahr'ej; der Zeiger ist der Content'go; Wir fügen ein neues Element ein. - Zeiger- qgt'zeiger
```

Die Frage/Leere erhalten (Leere) mit einem Strukturvorbehalt: ? ? ? ungültige Eingabe (leer) mit einer Reservierung bei der Strukturierung (neu; char /nMAX,vn-MAX); if (neu) (Reservierung) malloc (Größe (Struktur)))) printf (Name.....:); Fgets (n, MAX, stdin); strcpy (neuer Name, Zeichenfolge (n); printf (Vorname.:); Fgets (vn, MAX, stdin); strcpy (neuer Name, vn); printf (%d, neue Nummer); neu->gt;erwarteNULL; Put (neu); void (Leerheit) wenn (dummy- - NULL) - printf (%s, %s number.: % d , mannequin---re-name, dummy--gt;predicatable-->yard name, dummy---zeit;nummer); Holen (); sortier_eingeben (n,vn,at,am,aj,et,em,aj,ej,geh); pop(); printf (( ) printf⁢ENTER>(); getchar (); - int main () - int wahl; char dname (MAX); FILE - ; структуры name «пустой рекорд»; ; printf (Vipr->array[i]); ; daten (Leerheit) - Vin_t Vipr- LEN-sizeof (ind)); Vipr und LEn; für (int i = 0; i < Vipr->array[i]; i++) printf (Vipr->array[%d: %3d, i,Vipr->array); Rückgabe EXIT_SUCCESS; Die Initialisierung flexibler Elemente ist nach dem C99-Standard nicht zulässig. Ein Beispiel dafür ist: C99 Vint_t Viarr1 Nr. 4, 1, 2, 4, 8 sind gemäß der C99-Norm nicht erlaubt: Was ist ihre Meinung Wie gefällt Ihnen Opencbook? Wir freuen uns immer auf Ihre Anhörung. Bitte senden Sie uns Ihr Feedback per E-Mail kommunikation@rheinwerk-verlag.de. Seite 6 Der Zweck dieses Kapitels ist nicht, Ihnen eine Reihe von Algorithmen vorzustellen, sondern Ihnen einige Grundlagen der Informatik. Um an diesem Thema teilnehmen zu können, benötigen Sie einige Erfahrung in der Programmierung mit C. Sie müssen bereits alle Grundlagen von C kennen. Zunächst müssen Sie wissen, was Arrays von Kettenlisten sind und wie sie in der Praxis verwendet werden. Wenn Sie also einige dieser Techniken nicht vollständig verstehen oder vermissen, dann können Sie mit diesem Kapitel beginnen. Auch ein wenig Selbstmotivation ist nötig, um die Themen zu verstehen - und vor allem die Praxis. Wenn Sie dieses Kapitel gelesen und Algorithmen (hoffentlich) verwendet haben, dann haben Sie eine gute Grundlage, um in dieses Problem einzutauchen. 22.1 Was sind Algorithmen? Der Algorithmus hat keine Verbindung zum Bertriebssystem und ist nicht von einer Bibliothek abhängig. Ein Algorithmus ist nur eine Methode, die verwendet wird, um ein Problem unter bestimmten Bedingungen durch die endliche Anzahl von Schritten zu lösen. Mit dem Verfahren meine ich natürlich den Quellcode. Da es viele verschiedene Algorithmen gibt, ist es nicht immer einfach, den richtigen zur richtigen Zeit zu verwenden. Dies hängt vom Problemfall und der Erfahrung eines Programmierers mit einem bestimmten Algorithmus ab. Meistens ist es sinnvoll, verschiedene Algorithmen zu testen und eine Laufzeitanalyse (Profilierung) zu erstellen, insbesondere wenn Sie den Algorithmus nicht kennen und nicht wissen, was er erfordert. Was ist Ihre Meinung Wie gefällt Ihnen Opencbook? Wir freuen uns immer auf Ihre Anhörung Bitte senden Sie uns Ihr Feedback per E-Mail kommunikation@rheinwerk-verlag.de. Seite 7 Ob Sie jetzt Arrays oder Kettenlisten verwenden, Algorithmen in den folgenden Abschnitten können in der Regel mit mehreren Einstellungen beim Debuggen des Quellcodes zu halten, müssen Sie die Stack-Überprüfung zumindest beim Debuggen des Quellcodes und Stacks für Größe und Position einkontrollieren. 22.10 Zeigen zu vergrößern. die sich einander ähneln. Wieder, während (prt q) (prt1) Die Bedingung für den prt1-Index wird bereits nach den ersten Schritten (100'it;60) und, der zweite, zeigen, wird gar, nicht dekrementiert (70'gt;100) angewendet. Dies führt zu folgendem Zustand: Klicken Sie hier, um Abbildung 22.11 erneut zu vergrößern, zwei Werte wurden identifiziert, wobei prt1 nicht wenig er ist und prt2 nicht größer als x ist. Jetzt werden beide Elemente erneut ausgetauscht: Klicken Sie hier, um Abbildung 22.12 prt1 und prt2 nach der Freigabe des Wertes zu vergrößern. Beide Hände bewegen sich dann zueinander, so dass das nächste Bild nun angezeigt wird (siehe Abbildung 22.13). Klicken Sie hier, um das Bild von Abbildung 22.13 prt1 und prt2 zu vergrößern. Nach zwei Zeilen während (prt2)x) prz---; wenn die Bedingung (prt1 zgt; prt2) jetzt wahr ist und Pausen tun, während der Zyklus intermittierend ist. Der folgende Zustand ist verfügbar (siehe Abbildung 22.14). Klicken Sie hier, um das Bild von Abbildung 22.14 prt1 und prt2 vergrößern. Damit wurde der Prozess der ersten Trennung beendet. Es zeigt auch, dass alles links von der Trennlinie größer als alles rechts davon kleiner ist. Der Algorithmus funktioniert auch, wenn der Zustand ist nicht von einer Bibliothek vor dem Pop() velient. Um die Stack-Überprüfung zumindest beim Debuggen des Quellcodes zu halten, müssen Sie die Anzahl der Stöße und Stacks zählen und entsprechend reagieren. Ein möglichen Beispiel könnte wie folgt aussehen. count_push_pop.c. #include <define <stdio.h> DEBUG int push_cnt'0, pop_cnt'0; #endif void push (void pop_cnt; printf (Zahlenschub : %d, push_cnt); #endif / Push s/ printf (tnumber pop : %d, pop_cnt); wenn (pop_cnt push_cnt) printf (Schwerer Fehler: pop_cnt sollte nie mehr als push_cnt sein); #endif / - Ausführen einer Pop-Version (Leerheit) s Push (); Pop Pop Pop Pop EXIT_SUCCESS Pop:; Was ist Ihre Meinung Wie gefällt Ihnen Opencbook? Wir freuen uns immer auf Ihre Anhörung. Bitte senden Sie uns Ihr Feedback per E-Mail kommunikation@rheinwerk-verlag.de. Seite 4 Ein weiterer Typ abstrakter Datenstrukturen sind Warteschlangen. Die Warteschlangen können als Wartschlange an der Kasse dargestellt werden. Die erste Person, die diese Position einnimmt, ist auch die erste. Alle anderen Kunden sollen immer hinter stehen und warten ,, wenn sie es sind). Warteschlangenoperationen (Folienelement und Elementausgabe) werden Put und Get genannt. Im Gegensatz zum Stapel werden sie Elemente in der gleichen Reihenfolge angezeigt, in der sie eingefügt wurden. Daher wird die Warteschlange auch als die erste Pipe vorstellen, die an beiden Enden Größe öffnet ist. Neue Elemente werden an einem Ende verschoben, und neue Elemente werden an anderen entfernt. Wie der Stapel hat die Warteschlange zwei Hauptfunktionen: get -- ein neues Element, das an das Ende der Warteschlange angefügt ist. put () - Der Artikel wird am anderen Ende entfernt. Abbildung 21.48 veranschaulicht das Prinzip der Warteschlange: Klicken Sie hier, um Abbildung 21.48 Warteschlange zu vergrößern, und ihre Features erhalten () und setzen () in der Praxis können Sie die Warteschlange verwenden. Eine interessante Lösung für die umfangreiche Liste, die Sie in Abschnitt 21.3 erstellt haben, wäre die Verwendung der Warteschlange zum Speichern der Daten. Dies ist sinnvoll, insbesondere in einem System, in dem mehrere Benutzer gleichzeitig auf die Daten zugreifen sollen - vor allem, wenn mehrere Benutzer versuchen, gleichzeitig in dieselbe Datei zu schreiben. <stdlib.h></stdio.h>Warteschlangen können Sie sogenannten Sackgassen vermeiden. Dummheiten sind klassische Synchronisierungsprobleme, die auftreten, wenn mehrere Prozesse über dieselbe Ressource streiten und sich nicht mehr entwickeln. Keine Sorge, ich werde dieses Problem hier nicht noch einmal recht umfangreiche Liste erweitern. Wenn Sie möchten, können Sie den Suchbegriff Warteschlange C einmal in der Suchmaschine eingeben, einschließlich Betriebssystem. Szenario: Damit der Arzt einen Überblick darüber erhält, ob sich noch Patienten im Wartezimmer befinden und vor allem welcher Patient als nächstes da ist, muss das Programm geschrieben werden. Die Daten des neuen Patienten werden vom Assistenten bei der Terminvereinbarung aufgehängt. > >-schlange_init- . . . . . i >--NULL;

Rückgabe 0; Es wird auch als Kopf, eine Art Dummy verwendet. Es zeigt immer den Anfang der Warteschlange an. Zuerst wird der Platz für den Dummy reserviert und dann mit bedeutungslosen Werten initiiert. Der vorherige Zeiger zeigt also wieder auf NULL am Anfang: Klicken Sie hier, um abbildung 21.49 in der leeren Warteschlange weiter zu vergrößern, was immer ein neues Element am Ende der Warteschlange aufhängt: int put (Struktur neuer Reservierung) / - ist das erste Element in der Warteschlange? Die Fragewörter (dummy-----null) s/Dies ist das erste Element der qgt. B/ immer noch redlicher Zeiger; / Wir suchen das Schlange, während (seiger-zgt; vorweggenommen! - NULL) zeger-zeger-zgt; Der Zeiger ist vorherschar; 1. vorweggenommenNULL; Rückgabe 1; Zuerst müssen Sie überprüfen, ob sich ein Element am Anfang der Warteschlange befindet. Dies ist am Anfang nicht der Fall, und so wird hinten ein neues Element hinzugefügt: Das neue Element zeigt derzeit nichts dahinter an. Hier sehen Sie den aktuellen Status: Klicken Sie hier, um Abbildung 21.50 zu vergrößern. Wenn der Dummy bereits auf das Element hinter ihnen zeigt, die !ndexetztrappe; während (Zeiger--gt;erwartet' NULL) Zeiger-Zeiger--000 Gehen Sie durch die Warteschlange von front bis end bis zum Ende der Warteschlange (NULL). Dann, wie das erste Element, das Sie eine Funktionsfunktion Entfernen Sie das Element in der Warteschlange, und erstellen Sie zuerst, es gibt keine Patienten in Warteraum .....;; sehen Sie die Funktion dafür: Leere zu erhalten (Leere) s struct-Klausel Zeiger; / / Gibt es überhaupt etwas in der Wartescblange? <stdlib.h></stdio.h></time.h> </=> </stdlib.h> </time.h> </stdio.h> </stdio.h> • void shellsort (int *array, int elements) - int i, j, temp, n; /* Berechnen Sie den Anfangswert für den Abstand (x) prz--; wenn (my_qsort)-Links my_qsort (prt1, rechts) > (prt1 > prt2) wäl;/=> </stdlib.h> </time.h> </stdio.h> </stdio.h> • void shellsort (int *array, int elements) - int i, j, temp; /* Loop from left-1 to the right */ for(index=1; index<! ;index=elements; index++)= { = aktuelle= position= zwischenspeichern= */= wert_klein=array[index]; klieneren= wert= als= wert_klein= suchen.= schleife= läuft= *= von= aktueller= position= von= insertion(int *array, int index,wert_klein, wert_klein) - int index,index_klein,wert_klein; /* Eintritt in die Schleife, um das Bild von Abbildung 22.9 Swap-Werte prt1 und prt2 jetzt zu vergrößern, bewegen sich die beiden Zeiger miteinander. Klicken Sie hier, um das Bild von Abbildung 22.10 Zeigen zu vergrößern, die sich einander nähern. Wieder, während (prt1 q) (prt1) Die Bedingung für den prt1-Index wird bereits nach den ersten Schritten (100'it;60) und, der zweite, zeigen, wird gar, nicht dekrementiert (70'gt;100) angewendet. Dies führt zu folgendem Zustand: Klicken Sie hier, um Abbildung 22.11 erneut zu vergrößern, zwei Werte wurden identifiziert, wobei prt1 nicht wenig er ist und prt2 nicht größer als x ist. Jetzt werden beide Elemente erneut ausgetauscht: Klicken Sie hier, um Abbildung 22.12 prt1 und prt2 nach der Freigabe des Wertes zu vergrößern. Beide Hände bewegen sich dann zueinander, so dass das nächste Bild nun angezeigt wird (siehe Abbildung 22.13). Klicken Sie hier, um das Bild von Abbildung 22.13 prt1 und prt2 zu vergrößern. Nach zwei Zeilen während (prt2)x) prz---; wenn die Bedingung (prt1 zgt; prt2) jetzt wahr ist und Pausen tun, während der Zyklus intermittierend ist. Der folgende Zustand ist verfügbar (siehe Abbildung 22.14). Klicken Sie hier, um das Bild von Abbildung 22.14 prt1 und prt2 vergrößern. Damit wurde der Prozess der ersten Trennung beendet. Es zeigt auch, dass alles links von der Trennlinie größer als alles rechts davon kleiner ist. Der Algorithmus funktioniert auch, wenn der Zustand ist nicht von einer Bibliothek abhängig. Ein Algorithmus ist nur eine Methode, die verwendet wird, um ein Problem unter bestimmten Bedingungen durch die endliche Anzahl von Schritten zu lösen. Mit dem Verfahren meine ich natürlich den Quellcode. Da es viele verschiedene Algorithmen gibt, ist es nicht immer einfach, den richtigen zur richtigen Zeit zu verwenden. Dies hängt vom Problemfall und der Erfahrung eines Programmierers mit einem bestimmten Algorithmus ab. Meistens ist es sinnvoll, verschiedene Algorithmen zu testen und eine Laufzeitanalyse (Profilierung) zu erstellen, insbesondere wenn Sie den Algorithmus nicht kennen und nicht wissen, was er erfordert. Was ist Ihre Meinung Wie gefällt Ihnen Opencbook? Wir freuen uns immer auf Ihre Anhörung. Bitte senden Sie uns Ihr Feedback per E-Mail kommunikation@rheinwerk-verlag.de. Seite 7 Ob Sie jetzt Arrays oder Kettenlisten verwenden, Algorithmen in den folgenden Abschnitten können in der Regel mit mehreren Einstellungen beim Debuggen des Quellcodes zu halten, müssen Sie die Anzahl der Stöße und Stacks zählen und entsprechend reagieren. Ein möglichen Beispiel könnte wie folgt aussehen. count_push_pop.c. #include <define <stdio.h> DEBUG int push_cnt'0, pop_cnt'0; #endif void push (void pop_cnt; printf (Zahlenschub : %d, push_cnt); #endif / Push s/ printf (tnumber pop : %d, pop_cnt); wenn (pop_cnt push_cnt) printf (Schwerer Fehler: pop_cnt sollte nie mehr als push_cnt sein); #endif / - Ausführen einer Pop-Version (Leerheit) s Push (); Pop Pop Pop Pop EXIT_SUCCESS Pop:; Was ist Ihre Meinung Wie gefällt Ihnen Opencbook? Wir freuen uns immer auf Ihre Anhörung. Bitte senden Sie uns Ihr Feedback per E-Mail kommunikation@rheinwerk-verlag.de. Seite 4 Ein weiterer Typ abstrakter Datenstrukturen sind Warteschlangen. Die Warteschlangen können als Wartschlange an der Kasse dargestellt werden. Die erste Person, die diese Position einnimmt, ist auch die erste. Alle anderen Kunden sollen immer hinten stehen und warten ,, wenn sie es sind). Warteschlangenoperationen (Folienelement und Elementausgabe) werden Put und Get genannt. Im Gegensatz zum Stapel werden sie Elemente in der gleichen Reihenfolge angezeigt, in der sie eingefügt wurden. Daher wird die Warteschlange auch als die erste Pipe vorstellen, die an beiden Enden Größe öffnet ist. Neue Elemente werden an einem Ende verschoben, und neue Elemente werden an anderen entfernt. Wie der Stapel hat die Warteschlange zwei Hauptfunktionen: get -- ein neues Element, das an das Ende der Warteschlange angefügt ist. put () - Der Artikel wird am anderen Ende entfernt. Abbildung 21.48 veranschaulicht das Prinzip der Warteschlange: Klicken Sie hier, um Abbildung 21.48 Warteschlange zu vergrößern, und ihre Features erhalten () und setzen () in der Praxis können Sie die Warteschlange verwenden. Eine interessante Lösung für die umfangreiche Liste, die Sie in Abschnitt 21.3 erstellt haben, wäre die Verwendung der Warteschlange zum Speichern der Daten. Dies ist sinnvoll, insbesondere in einem System, in dem mehrere Benutzer gleichzeitig auf die Daten zugreifen sollen - vor allem, wenn mehrere Benutzer versuchen, gleichzeitig in dieselbe Datei zu schreiben. <stdlib.h></stdio.h>Warteschlangen können Sie sogenannten Sackgassen vermeiden. Dummheiten sind klassische Synchronisierungsprobleme, die auftreten, wenn mehrere Prozesse über dieselbe Ressource streiten und sich nicht mehr entwickeln. Keine Sorge, ich werde dieses Problem hier nicht noch einmal recht umfangreiche Liste erweitern. Wenn Sie möchten, können Sie den Suchbegriff Warteschlange C einmal in der Suchmaschine eingeben, einschließlich Betriebssystem. Szenario: Damit der Arzt einen Überblick darüber erhält, ob sich noch Patienten im Wartezimmer befinden und vor allem welcher Patient als nächstes da ist, muss das Programm geschrieben werden. Die Daten des neuen Patienten werden vom Assistenten bei der Terminvereinbarung aufgehängt. > >-schlange_init- . . . . . i >--NULL;

Anfang der Warteschlange an. Zuerst wird der Platz für den Dummy reserviert und dann mit bedeutungslosen Werten initiiert. Der vorherige Zeiger zeigt also wieder auf NULL am Anfang: Klicken Sie hier, um abbildung 21.49 in der leeren Warteschlange weiter zu vergrößern, was immer ein neues Element am Ende der Warteschlange aufhängt: int put (Struktur neuer Reservierung) / - ist das erste Element in der Warteschlange? Die Fragewörter (dummy----null) s/Dies ist das erste Element der qgt. B/ immer noch redlicher Zeiger; / Wir suchen das Schlange, während (seiger-zgt; vorweggenommen! - NULL) zeger-zeger-zgt; Der Zeiger ist vorherschar; 1. vorweggenommenNULL; Rückgabe 1; Zuerst müssen Sie überprüfen, ob sich ein Element am Anfang der Warteschlange befindet. Dies ist am Anfang nicht der Fall, und so wird hinten ein neues Element hinzugefügt: Das neue Element zeigt derzeit nichts dahinter an. Hier sehen Sie den aktuellen Status: Klicken Sie hier, um Abbildung 21.50 zu vergrößern. Wenn der Dummy bereits auf das Element hinter ihnen zeigt, die !ndexetztrappe; während (Zeiger--gt;erwartet' NULL) Zeiger-Zeiger--000 Gehen Sie durch die Warteschlange von front bis end bis zum Ende der Warteschlange (NULL). Dann, wie das erste Element, das Sie eine Funktionsfunktion Entfernen Sie das Element in der Warteschlange, und erstellen Sie zuerst, es gibt keine Patienten in Wartezimmer .....;; sehen Sie die Funktion dafür: Leere zu erhalten (Leere) s struct-Klausel Zeiger; / / Gibt es überhaupt etwas in der Wartescblange? <stdlib.h></stdio.h></time.h> </=> </stdlib.h> </time.h> </stdio.h> </stdio.h> • void shellsort (int *array, int elements) - int i, j, temp, n; /* Berechnen Sie den Anfangswert für den Abstand (x) prz--; wenn (my_qsort)-Links my_qsort (prt1, rechts) > (prt1 > prt2) wäl

*/ für (n = 1; n 0; n /= 3) s für (i = n; i = und nicht > * . . . . . für= (;= n=>. . . . . größer= als= temp= und= nicht= elemente= *= und=>. . . . . Temp; for(i = 0; i < elemente;= i++)= s= mini=i; for(j=i+1; j=> <= elemente;= j++)= if(array[j]=></=> < array[mini])= mini=j; = temp=array[mini]; array[mini]=array[i]; array[i]=temp; =void= einfügung(int= *array,= int= elemente)= s= int= i,= j,= temp;= für(i=1; i=> <= elemente;= i++)= temp=array[i]; aktuelles= element= zwischensing= */= for(j=i; array[j-1]=>temp&& j > 0; j--) /* solange der Vorgänger größer ist als das * aktuelle Element in temp ... */ array[j] = array[j-1]; /* gespeichertes Element in neuer Position */ array[j]=temp; . . . . . int i, temp; while(elemente--) for(i = 1; i <= elemente;= i++)= if(array[i-1]=>array[i]) en temp=array[i]; array[i]=array[i-1]; array[i-1]=temp; s float int main(void) int i; int elemente=1000; Zeit; clock_t starten Sie , ... (log.txt,a, stdout); i'1; <=3; i++,= elemente*=10)- printf(sortieren= von= %d= elemente,elemente);= printf(%d.= versuch= := alle= %d= elemente= muessen= sortiert= werden=elemente);= selectionsort= */= init_test_array(= elemente;= start=clock(); selection(test_array,= elemente-1);= ende=clock(); zeit=(float)(ende-start) (float)clocks_per_sec;= printf(selectionsort:= %.2f= sekunden,zeit);= insertionsort= */= init_test_array(elemente);= start=clock(); insertion(test_array,= elemente-1);= ende=clock(); zeit=(float)(ende-start) (float)clocks_per_sec;= printf(insertionsort:= %.2f= sekunden,zeit);= bubblesort= */= init_test_array(elemente);= start=clock(); bubble(test_array,= elemente);= ende=clock(); zeit=(float)(ende-start) (float)clocks_per_sec;= printf(bubblesort= := %.2f= sekunden,zeit);= shellsort= */= init_test_array(elemente);= shellsort(test_array,= elemente-1);= ende=clock(); zeit=(float)(ende-start) (float)clocks_per_sec;= printf(shellsort= := %.2f= sekunden,zeit);= quicksort= */= if(elemente=></=3;> < init_test_array my_qsort(test_array,= test_array+elemente);= ende=clock(); zeit=(float)(ende-start) (float)clocks_per_sec;= printf(quicksort= := %.2f= sekunden,zeit);= qsort= aus= der= standard-bibliothek=&gt init_test_array; старт - часы (); qsort (test_array, b, Größe (int), cmp_integer); (Float)(Ende-Start) / (float)CLOCKS_PER_SEC; printf (qsort : .2f Sekunden;Zeit); Попытка создать уже отсортированный список . /* </stdlib.h></=><<><<><<<<><>><<<<<<<<<<>>>>>>>><><<<<<<<<<&l

Bächen gibt es mehrere andere Möglichkeiten, binäre Bäume zu implementieren, zum Beispiel, um den Durchgang eines Baumes zu erleichtern. Sie haben bereits den Durchgang von Bäumen in Vorbestellung und Umgehungsreihenfolge gesehen. Wenn Sie jedoch iterativ machen und den Lytraversing-Bypass-Baum nicht mehr wiederholen möchten, können Sie die Struktur des Zeigers auf den übergeordneten Knoten erweitern: Struktur binaer_knoten' char ort '255'; Unsigniert Int bitte; Struktur binaer_knoten (links; Struktur binaer_knoten) auf der rechten Seite; Struktur binaer_knoten Jetzt kann jeder Knoten seine Umgebung steuern. Dieser elterliche Zeiger macht es einfach, einen Baum zu überqueren, aber der Schreibaufwand für das Programm erhöht sich. Dies verlangsamt auch das Einfügen und Entfernen des Elements, da der Zeiger mehr verwaltet werden muss. Der Stammzeiger des übergeordneten Elements überträgt dagegen einen Nullzeiger. Streams (Streams) ermöglichen es Ihnen, den Baum noch schneller zu passieren. Denn anstatt zu prüfen, ob die linke oder rechte Seite des Knotens leer ist, was zu einer schlechteren Laufzeit führen kann, müssen Sie nur zwei zusätzliche Bits (Bitfelder) in die Struktur einfügen: Struktur binaer_knoten char ort; Unsigniert int bitte; struktur binaer_knoten (links; Struktur binaer_knoten (rechts; Unsigned linker_thread:1; unsigned rechter_thread:1; z.B. wenn es einen anderen Knoten auf der linken Seite des Knotens gibt, ein bisschen linker_tread 1. Wenn es keinen Knoten gibt, ist es 0. Natürlich garantiert es in der Regel keine bessere Laufzeit, da es davon abhängt, wie der Compiler die Bitfelder optimiert. leider, der negative Aspekt der binären Bäume, die Sie gerade getroffen haben. Bitte beachten Sie den Binärbaum in Abbildung 22.24. Klicken Sie hier, um das Bild von Abbildung 22.24 Degenerative Binärbaum zu vergrößern Dieses Beispiel ist eigentlich ein binärer Baum und nicht, wie Sie vielleicht erwarten, eine Kettenliste. Der Baum hat eine Tiefe von 4 und kann in der Regel für 15 Elemente (24-1). Es ist ein degenerierter Baum. Dies kann auftreten, wenn ein relativ kleiner oder großer Wert als Stamm des Baumes verwendet wird. Gewährt Ein Beispiel kann ein ungünstiger Fall sein, aber es kann auftreten. Es gibt Algorithmen, um degenerierte binäre Bäume zu kompensieren, aber sie sind nicht leicht zu verstehen. Aber was bedeutet es, im Sinne von Binärbäumen perfekt ausbalanciert zu sein? Der binäre Baum ist perfekt ausbalanciert, wenn die Höhen (oder Tiefen) der linken und rechten Unterseite die wichtigsten sind. Solche Bäume werden AVL-Bäume genannt. Um Probleme zu vermeiden, sollten Sie den Baum immer ausgewogen halten. Es bedeutet jedoch auch eine Erhöhung der Speichermenge. Wenn ein neues Element eingefügt oder das Element entfernt wird, muss möglicherweise die gesamte Struktur neu organisiert werden. Sie sollten diese Anstrengung jedoch nur unternehmen, wenn ein schnelleres Suchergebnis wirklich wichtig ist. Dieses schnellere Suchergebnis wird durch eine Verkürzung der Länge des Pfades erreicht.22.4.10 Algorithmen für Balanced Trees - ein Überblick über dieses Thema können ganze Bücher geschrieben werden. Einige zusätzliche Informationen sollten dennoch über binäre Bäume zur Verfügung gestellt werden. AVL Trees AVL Tree (AVL nach seinen Erfindern Adelson-Velski und Landis) ist die Suche nach einem Baum, dessen Knoten sich in der Höhe (Tiefe) von denen der Subs durch nicht mehr als 1, d.h. einen perfekt ausbalancierten Baum, unterscheiden. Wenn diese Bedingung gebrochen ist, muss die Ausgleichsfunktion ausgeführt werden. Die folgenden drei Bedingungen können auftreten (auch Balance-Grad genannt): Balance ist zgt; 0 - die rechte Unterstütze hat eine höhere Höhe als die linke. Die Balance des 0-links-Subtriams hat eine höhere Höhe als die rechte. Balance Nr. 0 - Subtriri haben die gleiche Höhe und sind optimal ausbalanciert. Dieser Zustand wird derzeit angestrebt. Um diese Verletzung zu bekämpfen, findet die sogenannte Rotation statt. Es ist notwendig, die rechte Drehung (beeinflusst die rechte Seite des Subtrib des Knotens) und die Linkshänder-Rotation (beeinflusst die linke Seite des Subtrib des Knotens) zu unterscheiden. Es gibt auch einfache und doppelte Rotationen. Die Tatsache, dass alle diese Operationen sehr rechenintensiv sind, ist leicht zu erkennen. B-Bäume (Bayer Bäume) Wenn Sie daran interessiert sind, Ihre eigene Datenbank zu programmieren, dann sollten Sie sich mit diesem Baum beschäftigen. Die B-Struktur wird durch eine variable Anzahl von Elementen (Blätter) pro Knoten an die Größe des Dateisystemblocks angepasst. Dies gewährleistet eine effiziente und optimale Geschwindigkeit in verschiedenen Systemen. Einzelne B-Baumknoten sind nicht immer belegt und variieren zwischen dem kleinsten und dem höchsten Knoten. Somit gibt es immer Raum für Strukturänderungen während Der Manipulationen (Einfügen, Löschen, Ändern ...) in die Datenbank. waren bei weitem nicht alle Algorithmen, die verwendet werden können, um ausgewogene Bäume zu erstellen. Hier sind die oberen 2-3-4 Bäume und rot-schwarze Bäume. Wir freuen uns immer auf Ihre Anhörung. Bitte senden Sie uns Ihr Feedback per E-Mail kommunikation@rheinwerk-verlag.de. Seite 10 Hashing ist eine bekannte Methode zum Suchen und Speichern von Datensätzen mithilfe einer Tabelle, die einen Schlüssel in eine Tabellenadresse konvertiert. Diese Konvertierung ermöglicht es Ihnen, zu dieser Tabellenadresse auf einmal zu gehen, während Sie suchen, ohne den gesamten Eintrag zu überqueren (Teil des Eintrags in der binären Suche). Der Zweck des Hashings besteht darin, die vorhandene Speicherkapazität besser zu nutzen und den Zugriff zu beschleunigen.22.5.1 Wann wird Hashing verwendet? Das beste Beispiel für Hashing sind Zeichentabellen, in denen die Werte mit dem dynamischen Satz von Zeilen jedes Elements verknüpft sind. Das passiert zum Beispiel mit dem Compiler. Dies ist die effektivste Möglichkeit für den Compiler, alle Variablen im Programm zu verwalten. Ein weiteres Beispiel ist die (verräterische) automatische Vervollständigung eines Browser- oder Webbrowsercache, in dem die Story gespeichert wird. Ein einfaches Beispiel für Hashing ist das Zählen der Wörter, die im Text vorkommen. Anforderungen, die auf ein digitales Wörterbuch angewendet werden, können genauso effektiv sein wie beim Programmieren großer Datenbanken. Denn in diesen Fällen ist es wirklich notwendig, eine unglaubliche Menge an Daten in kurzer Zeit zu finden.22.5.2 Was braucht es, um Hash? Hashfunktion - Hashfunktion erstellt eine eindeutige Adresse, die die Position der Aufzeichnung in der Hashtabelle bestimmt. Hashtabelle - Hashtabelle enthält eindeutige Adressen. Die Suchabfrage wird zuerst mithilfe der Hashfunktion in eine Tabellenadresse konvertiert. Diese eindeutige Adresse wird dann verwendet, um nach einem Datensatz in einer Hashtabelle zu suchen. In der Praxis wird die Hashfunktion in der Regel durch ein Array der entsprechenden Größe bestimmt, das während der Kompilierung erstellt wird. Einfach ausgedrückt, eine 10-Elemente-Hashtabelle ist ein Array von Kettenlisten, die 10 groß sind. Jeder Index dieses Arrays kann über eine Kettenliste verfügen. Die Angelegenheit sollte nun in der Praxis untersucht werden. Auch hier wird das Beispiel der Postleitzahl verwendet: um zu strukturieren bitte char ort-MAX; Unsignierte Int postleit; Bitte strukturieren Sie next; Die Hashtabelle, die derzeit für die Kettenliste verwendet wird, lautet wie folgt: Struktur, bitte, hash_tabelle MAX_HASH; Um es einfach zu halten, wird ein Array mit niedriger Größe verwendet: Abbildung 22.25 zeigt die aktuelle Hashtabelle in der Grafischen Ansicht: Hier Um das Bild zu vergrößern Abbildung 22.25 Leere Hashtabelle Sie benötigen jetzt ein Feature, um einen neuen Eintrag zum Hash hinzuzufügen: bitte einfügen (char o, unsigned int p)/ Hash-Wert (Bucket) in hash_adresse (0-9) (int hash_adresse und hash_funktion (o); / printf %d,hash_adresse)hash_tabelle; Der Zeiger für die Abrechnungsadresse der Tabelle hash_tabelle: hash_funktion der hash_adresse hash_adresse Ausgabe/Printf (%s mit Postleitzahl %d ist bereits vorhanden, o,p); Vermeiden Sie das Duplizieren von Datensätzen und umgekehrten Zeigern. 'pointer'pointer->next; To allogate'/) Speicher für das neue Element - der Wegweiser - malloc (sizeof (struct please)); Wenn (NO NULL) 'printf' kein Speicher für die neue Postleitzahl; O); Zeiger und p; Zeiger-hash_tabelle hash_adresse; hash_tabelle hash_adresse einen Zeiger; Umgekehrter Zeiger; Die Funktion wird nun Schritt für Schritt erklärt. Beginnen wir mit: int hash_adresse und hash_funktion(o); Dadurch wird eine Variable hash_adresse Hashwert, der logischerweise zwischen 0 und 9 liegen sollte, da die Hashtabelle über 10 Steckplätze verfügt. Angenommen, die Einfügefunktion wurde den folgenden Werten gegeben: einfügen (Augsburg, 86163); In diesem Beispiel berechnet die Hashfunktion 6 aus der Augsburger Zeile. Somit ist die Augsburger Linie im Index (Slot) von 6 Hash-Tabellen enthalten. Nur ein Zeiger sollte auf diesen Steckplatz verweisen: /Der Zeiger auf die Tabellenadresse, hash_funktion von hash_tabelle/hash_adresse; Optisch sieht es so aus: Klicken Sie hier, um auf Abbildung 22.26 Zeiger für den Abrechnungsindex zu zoomen Jetzt müssen Sie einen freien Platz für neue Daten finden: während die zgt; how (pointer ! question/ if (pointer-zgt;postcode q p) - Postleitzahlen gleich sind? Im ersten Fall ist es ziemlich trivial. p; pointer-zgt;and hash_tabelle hash_adresse; hash_tabelle hash_adresse Zeiger; jetzt ist das erste Element in der Hash-Tabelle. Grafisch dargestellt, sieht es wie folgt aus: Klicken Sie hier, um Abbildung 22.27 zu vergrößern. : Beilage (Friedberg, 86316); Die Hash-Funktion, die immer noch von Ihnen gehalten wird, berechnet index (slot) 8. Wenn Sie an der Einfügefunktion () arbeiten, klicken Sie hier, um Abbildung 22.28 zu vergrößern. Nun zurück zum konkreten Fall des Hashings. Folgende neue Datensätze werden eingeführt: Einfügung (Stuttgart, 70190); Die Hash-Funktion berechnet den Wert des Index 8 aus der Stuttgarter Linie. Wenn Sie sich Abbildung 22.29 ansehen, werden Sie sehen, dass Slot 8 bereits Inhalt hat (Friedberg). Dies wird als Kollision bezeichnet, was bedeutet, dass die Hashfunktion die gleichen Werte für verschiedene Schlüssel bereitstellen kann. Aus diesem Grund wurde eine lineare Kettenliste verwendet. So wird der neue Datensatz einfach nach Friedberg eingefügt. In der Informatik wird dies als Eine Kette von Synonymen bezeichnet. Klicken Sie hier, um das Bild von Abbildung 22.29 zu vergrößern Der Datensatz wurde nach der Kollision hinzugefügt.22.5.3 Hash-Funktion Mit diesem Beispiel sollte klar sein, dass die Hashfunktion eine entscheidende Rolle spielt. Wie schreiben Sie eine solche Funktion vor Ihnen. Sie können sogar ein Feature schreiben, das Zufallszahlen zwischen 0 und 9 zurückgibt. Aber was nützt eine solche Funktion, wenn 90 % der Zufallszahlen, z. B. zwischen 2 und 4, liegen? Der Rest der Slots ist praktisch ungenutzt. Es gibt drei Möglichkeiten, den Hash effektiv zu berechnen: die Abteilungsmethode und der Schlüssel mod mFor m Für m, idealerweise sollten Sie eine Primzahl so nah wie möglich am höchsten Index wählen. Die zentrale quadratische Methode Schlüssel II ist der Schlüssel, wo die führenden und endende Figuren entfernt werden müssen. Beispiel: H (3206) - 32062 - 10278436 Werte 10278436 schneiden abwechselnd die rechte und linke Ziffer, bis der Wert erstellt wird, kleiner als der Index. Wenn Sie beispielsweise eine Hashtabelle mit einem Index wert 10 angekündigt haben, sieht der Schlüsselwert 10278436 wie folgt aus: Die methode der menschlichen Zersetzung zersetzt den Schlüssel, bis er eine gültige Adresse hat. Als Beispiel verwenden wir den Schlüssel 135612: 135612, 81 x 81, der Schlüssel wurde zersetzt, bis er einen Wert von 9 als gültige Adresse haben würde. Hashing der Zeichenfolgen Ein bewährter Hashalgorithmus für Zeichenfolgen generiert einen Hashwert, in dem jede Zeichenfolge mehreren Zeilen hinzugefügt wird. Beim Multiplizieren werden einzelne Bits auf einen zuvor berechneten Wert verteilt. Tests zeigten, dass für die Linien 31 und 37 als gute Multiplikatoren dargestellt wurden, die auch für Z.B. Programme verwendet werden. Theoretisch können Sie sich ein Beispiel mit der POSTcode-Liste vorstellen. Die Postleitzahlen 70190 wurden für die Stuttgarter Zeichenfolge eingeführt: hash_tabelle Stuttgart - 70190; Tatsächlich handelt es sich um ein Array mit einem Indexwert als Zeichenfolge. In der Praxis ist es sicherlich nicht Вы также пишете функцию хэша для этого. . . . . Nr. hash_funktion (char s string) - int hash_adresse; Char-Zeiger; hash_adresse Nr. 0; - (-zeichen) в то время как (указатель! ' "MAX_HASH hash_adresse hash_adresse hash_adresse') , ... Относительно трудно найти оптимальную функцию хэширования. В таком случае вам нужно будет проверить, пока вы не удовлетворены результатом. Hier sehen Sie das komplette Programm, um das Hashing mit separater Verkettung zu demonstrieren: /* hashing.c */ #include <stdio.h>#include #include #include #define MAX_HASH <string.h> <stdlib.h>10 #define M 31 struct please' char ort[255]; nicht signiert int postleit; struct bitte *next; s; struct bitte *hash_tabelle[MAX_HASH]; struct bitte *hash_tabelle nicht signiert int); void search_in_hash(char *); int hash_funktion(char *); struct bitte *insert(char *o, unsigned int p) s struct bitte *pointer; /* Hashwert (Bucket) zu hash_adresse (0-9) */ int hash_adresse = hash_funktion(o); /* printf(%%d,hash_adresse); */ /* Zeiger auf berechnete Tabellenadresse * durch hash_funktion */ Zeiger = hash_tabelle[hash_adresse]; /* Wir suchen nach freiem Speicherplatz für einen neuen Eintrag * in hash_tabelle[hash_adresse] */ while(Zeiger != NULL) - if(strcmp(o, pointer->ort) == 0) /* City equal? */ if(pointer->postleitzahl == p) * =* Postleitzahlen gleich? */ printf(%s mit Postleitzahl %d ist bereits vorhanden ,o,p); /- Избегайте дублирования записей и обратного указателя; • . . . . . ( ) - printf () () NULL; s strcpy (a->ort, o); > . . . . . • hash_tabelle hash_adresse; hash_tabelle hash_adresse ; обратный указатель; Nr./search_in_hash (char q o) s struct, hash_adresse hash_funktion . . . . . - NULL) - (strcmp (->ort, o) Nr. 0) printf (€€€%ddo,pointer->postleit); . . . . . / - Nr. / int hash_funktion (char s string) - int hash_adresse; Char-Zeiger; hash_adresse Nr. 0; - (-) (-zeichen) ()zeichenfolge; в то время как (указатель! ' "0'hash_adresse)</stdlib.h> </stdio.h> </stdio.h> Zeiger; Indexpost; Hash_adresse % MAX_HASH-ertrag; int haupt (leer) Einsatz Nr./Einsatz (Friedberg, 86316); Einsatz (Augsburg, 86136); Einsatz (Stuttgart, 71345); Search No/ search_in_hash (Augsburg); search_in_hash (Friedberg); search_in_hash (Stuttgart); EXIT_SUCCESS; Die search_in_hash Suchfunktion ähnelt der Einfügung. Ich erkläre es also nicht weiter. Es ist jedoch auch wichtig zu beachten (auch wenn es logisch sein sollte), dass jedes Feature, das Sie hinzufügen (Suchen, Sortieren, Löschen, Einfügen, ...) die gleiche Hashfunktion verwenden sollte.22.5.4 Hashing mit einer direkten Adresse kann auch von jeder Hashe direkt gelöst werden, vorausgesetzt, Sie können abschätzen, wie viele Elemente eingefügt werden. Dies basiert auf einer Tabelle, die nur Zeiger auf andere Stellen im Speicher gespeicherter Datensätze enthält. Direkter Kontakt kann wie folgt hergestellt werden (Pseudocode): während (Schlüssel_stimmt_nicht_überein) - wenn (Schlüssel_stimmt_überein) - printf (gefunden); Rückgabe sonst wenn) - printf (nicht gefunden); Rückgabe weiter_an_die_nächste_Position; Der Vorteil der direkten Zirkulation liegt in mehr Geschwindigkeit. Der große Nachteil ist jedoch die feste Größe des Tisches. Wenn Sie die Datenmenge schätzen können, ist dies kein Nachteil. Für Datenbanken, in denen Daten von Fall-zu-Nutzung abhängig sind, macht die direkte Zirkulation keinen Sinn.22.5.5 Vergleich von Hashing mit binären Bäumen Vorteile des Hashings: Es ist leicht, die Vorteile von Binärbäumen zu realisieren: Leistungsgarantie unterstützt auch im schlimmsten Fall viele andere Operationen (z.B. Sortieren) dynamisch (im Gegensatz zu Binärbäumen sind keine Informationen über die Anzahl der Stände erforderlich).) Was ist Ihre Meinung Wie gefällt Ihnen Openbook? Wir freuen uns immer auf Ihre Anhörung. Bitte senden Sie uns Ihr Feedback per E-Mail kommunikation@rheinwerk-verlag.de. Seite 11 Word-Prozessoren verwenden Zeilen in Form von einzelnen Buchstaben, Zahlen oder Sonderzeichen, um Text zu bearbeiten. Wenn Sie einen Texteditor oder ähnliches entwickeln möchten, benötigen Sie auch Funktionen wie Zeilensuche oder Subladung; oder z. B. eine Syntax, die eine bestimmte Programmiersprache hervorhebt. Eine weitere Möglichkeit besteht darin, Muster zuzuordnen, die Sie möglicherweise von Perl- oder Linux-Shells kennen. Für diese und andere Anwendungen werden Linienabgleichsalgorithmen verwendet. Sie arbeiten nach folgendem Prinzip: In einer Textzeile, wie z. B. dem Text in diesem Buch, möchten Sie die Häufigkeit von N-Zeichen und M-Zeichen vergleichen, die in der Suchvorlage enthalten sind. Der Zweck des Kapitels besteht nicht darin, Algorithmen mit mathematischen Formeln zu erklären, sondern Algorithmen so zu erklären, dass Sie das Prinzip verstehen, nach dem sie funktionieren. Die Schnittstelle dieser Beispiele besteht darin, eine Struktur zu verwenden, die Daten aus einer Befehlszeile aufnimmt und für mögliche Schätzungen speichert. struct file-char Name LEN; / Dateiname, int gefunden; / Nummer gefunden typedef FILE Strukturierungsdatei; Sie können diese Struktur um zusätzliche Informationen erweitern, z. B. Standort. In Beispielen wird ein Array von Strukturen gleichzeitig verwendet. In der Praxis können Sie auch Kettenlisten verwenden. Aufruf an Programme immer: der Name des Programms suchstring file1 ... Bevor datei_n für alle Add-Ons, müssen Sie sich noch auf einzelne Algorithmen konzentrieren. Alle relevanten Algorithmen suchen nach einer bestimmten Textsequenz. Wenn Sie an ganzen Wörtern interessiert sind, können Sie den Algorithmus entsprechend einrichten. Sie sollten sicherstellen, dass alle Whitespace-Zeichen (Newline, Tab und Space) vor und nach der Suchzeile beobachtet werden.22.6.1 Brute Force Algorithm Der einfachste Algorithmus ist offensichtlich. Alle möglichen Bestimmungen der Vorlage im Text werden überprüft, bis die Vorlage mit dem Text übereinstimmt oder der Text endet. Das vollständige Muster wird beim Textvergleich um eine Position voraus gezählt. Es ist ein Brute-Force-Algorithmus (oder ein grober Algorithmus oder ein naiver Algorithmus). Abbildung 22.30 zeigt ein einfaches Beispiel. Klicken Sie hier, um die Abbildung 22.30 Sequenz des Brustod-Algorithmus zu vergrößern. Hier sehen Sie den Quellcode für diesen einfachen Zeilenabgleichsalgorithmus: / bruteforce.c qlt.h'gt; #include #include #include #define LEN'lt.string. #define MAX_DAT 10 #define MAX_LINE 4096 #define LINE ---------- struct file char name'LEN; / / Dateiname / - Die Anzahl der gefundenen q/ s gefunden; typedef struct datei FILE; int i - Textzählerebene - int j - q/ int BruteForce Counter Level (ball, char s text) int i q 0, j, cnt 0; int m'strlen (Muster); MAX_DAT // Длина <=n-m) {= solange= i= kleiner= als= n-m= */= solange= muster= und= text= gleich= j++= */= for(j=0;></=n-m)> <m &&= muster[j]==text[i+j]; j++);= if(j==m) {= ist= die= länge= von= j= gleich= der= vom= muster?= */= printf(pos.= %3d,= ,i);= cnt++;= }= i++;= im= text= eine= position= weiter= */= }= return= cnt;= }= int= main(int= argc,= char= *argv[])= {= datei= suche[max_dat];= char= suchstring[len];= char= read_line[max_line];= file= *f;= int= i,= j= ,= ret,= zeile;= if(argc=></m> < 3)= {= fprintf(stderr,= verwendung:= %s= suchstring= datei1= => <datei2> <datei%d>шаблона возвращение EXIT_FAILURE; strncpy (такие стринги, argv,</datei%d> </datei2> </stdlib.h> </string.h> </stdio.h>/- Bewerten Sie die Argumente der Befehlszeile für (i'2,j'0; j zlt; MAX_DAT Ich bin argc; i',j') argv'i, LEN); search'j'.found - 0; s für (i q 0; i zlt; argc-2; i') f - fopen (suche, r); if (f q NULL) - perror (null); s Zeile 0; printf (file% while (fgets (read_line, MAX_LINE, f) s zeile'; ret - BruteForce (suchstring, read_line); if (retired! - 0) s search'i'.found'ret; printf (in zeile %d,line); ret No 0; .printf (such ergebnisse% d search. printf (LINE); fclose (f); - return EXIT_SUCCESS; Wenn wir die ex-Suchleiste als Beispiel und als Vorlage ein Textbeispiel als Beispiel verwenden, wird der interne Zyklus nur dreimal inkrementell, jedes Aussehen des Buchstabens e. Das Ergebnis wird zweimal gefunden. Kraft hat immer lineares Verhalten der Zeit.22.6.2 Knut/Morris/Pratt Algorithm (KMP) Das Fehlen eines Brute-Force-Algorithmus ist, dass es hartnäckige Zeichen nach Zeichen, Position vergleicht. , Morris und Pratt, nach denen dieser Algorithmus auch benannt ist, haben diesen Algorithmus verbessert (geklärt). Sie hatten die Idee, Inkonsistenzen in einen weiteren Algorithmus aufzunehmen. Ein Beispiel ist diese Textfolge (Text): lu lalalala lule lulalalas Suchleiste (Suchmuster) ist aalas. Der Prozess beginnt als Brute-Force-Algorithmus: Klicken Sie hier, um Abbildung 22.31 auf der Suche nach der Suchleiste Alalas zu vergrößern - Starten Sie hier, sie haben keine Übereinstimmung zwischen Text und Suchmuster. So können Suchmuster um eine Position weiter geschoben werden: Klicken Sie hier, um Abbildung 22.32 No Match zu vergrößern - die Suchleiste bewegt sich um eine Position weiter. Dies wird wiederholt, bis die beiden identischen Zeichen aufeinandertreffen: Klicken Sie hier, um das Bild von Abbildung 22.33 zu vergrößern. Bis der Text nun mit dem Suchmuster identisch ist, sind ich und j Inkremente: Klicken Sie hier, um Abbildung 22.34 zu vergrößern. Hier gibt es Ungleichheit in Textbestimmungen und Suchmustern. Mit einem Brute-Force-Algorithmus wird die Vorlage nun um eine Position nach vorne ausgesetzt. Und hier greift Knut, Morris und Pratts Algorithmus ein. Die geringste mögliche Bewegung, in der der Aalas mit sich selbst zusammenfällt, betrifft zwei Stellen: Klicken Sie hier, um das Bild bild22.35 zu vergrößern. sich nicht schneiden. Genauer gesagt, sie wissen es jetzt, weil ich es Ihnen gesagt habe, aber Sie wissen nicht, wie es durch Software geschehen wird. Dies erfordert eine Vorphase. Aber mehr dazu. Klicken Sie hier, um Abbildung 22.36 zu vergrößern, um auf zwei Ziffern statt auf eine zu zoomen. Theoretisch klingt das natürlich alles ziemlich interessant. Aber in der Praxis ist es viel komplizierter. Wie verstehen Sie die geringste Bewegung um die Suchleiste (Suchmuster) selbst? Da die Berechnung dieser Verschiebung nicht vom Text abhängt, in dem die Suche stattfindet, sondern nur vom Suchtext (dem Suchmuster), kann sie vor der eigentlichen Suche erstellt werden. Es wird die Vorverarbeitungsphase genannt. Hier sehen Sie einen Algorithmus, der eine Sprungtabelle aus der Suchleiste selbst für den eigentlichen Suchalgorithmus erstellt: ungültige init_next (Suchmuster, int m) j; i - 0; j - nächste '0' -1; /' solange ich kleiner als eine Suchleiste bin, während (i'lt; m) s während (j'gt; -1 ' Suchmuster 'i'i'i'j'j'j') j Um zum Skript zurückzukehren, verwenden Sie einen Brute-Force-Algorithmus und vergleichen einzelne Zeichen. Wenn es eine Diskrepanz (Suchmuster! - Text) gibt, wird die Zeile mit der maximalen Länge (L) von hinten in zuvor gefundenen und entsprechenden Zeichen durchsucht, was auch der Anfang einer anderen Zeile ist. Das Zeichen wird dann mit L-1 (nextj) und dem ito-Symbol im Text verglichen. Es gibt zwei Möglichkeiten, dies zu tun: der Charakter der nächsten Suchzeile entspricht der Ethan-Natur des Textes. So wird entweder so normal wie beim Brute-Force-Algorithmus fortgesetzt, bis die gesamte Linie gefunden wird oder die Diskrepanz wieder auftritt. In diesem Fall gilt dasselbe wie bei der ersten Diskrepanz. Das nächste Suchleistensymbol entspricht nicht der Ethan-Natur des Textes. der nextj wird somit um den Wert 1 verringert, und der Vergleich wird fortgesetzt. Wenn es eine weitere Diskrepanz gibt, wird die nächste verschwommen, bis das erste Suchleistensymbol erreicht ist. In diesem Fall bin ich in einem Schritt und alles beginnt von vorn. Stellen Sie mit dem folgenden j sicher, dass der j-1 die Länge des größten Endes des Stückes, aber auch das Originalstück der Suchleiste ist. Wenn das Suchmuster dem Suchmuster entspricht, wird der folgende Wert zugewiesen, da das nächste Array immer das folgende Zeichen enthält. Ist dies nicht der Fall, werden der Anfangsteil und der längste Teil des Endes mit dem Muster verglichen, bis ein positiver Vergleich zwischen dem Muster und dem Muster vorgenommen wird. Nun sollte der eigentliche Algorithmus mit einer Suchzeile und einem soeben verwendeten Textbeispiel /' kmp.c q/ #include <stdio.h>#include #include #define MAX <stdlib.h> <string.h>4096 init_next (char, int); kmpSearch (char, char); int next 'MAX); / i - im Text j'0; int m'strlen (muster); L'nge Muster (int n'strlen (); / / L'nge Text (init_next (muster, m); / Tabelle f'r next berechnen s/ s/ s/ (i<n) init_next - printf (Gefunden an Pos. %d, i-j); j - -1; /- kleiner als der Suchstring ist , ... (см. Я - соберусь) ? ()) : (»i'j); - Int main () - kmpSearch (, lu lalalala lule lulalalas); EXIT_SUCCESS; Das vollst'ndige ,,Brute-Force-Algorithmus umgeschrieben auf dem Programmbeispiel: / / kmpsearch.c #include <stdio.h>#include #include #define LEN <stdlib.h> <string.h>255 #define MAX_DAT 10 #define MAX_LINE 4096 #define MAX 255 #define LINE /- , , ,

конечном итоге не найдет выход, хотя в теории есть еще некоторые – другими словами, Мистер C вращается кругами. Чтобы обойти эту проблему, вы можете либо немного усложнить исходный код, либо добавить другие навыки Мистеру C. Для этого, например, мистер C может in diagonale Richtungen bewegen. Klicken Sie hier, um das Bild von Abbildung 22.39 für mehr Freiheit zu vergrößern. Die Bewegung für Herrn C, so dass Sie jetzt die folgenden vier neuen Bewegungen haben, die Sie in den Code aufnehmen müssten: die rechte Höhe (x-1,y'1) (x-1,y'1) (x-1,y'1) (x-1,y'1) (x-1,y'1) (x-1,y'1) (x-1,y'1) (x-1,y'1) (x-1,y'1) (x-1,y'1) (x-1,y'1) (x-1,y'1) Linksunten (x-1,y-1) links hoch (x-1,y-1) Als nächstes ist es wichtig, noch einmal zu bestimmen, in welcher Reihenfolge diese (jetzt acht) Bewegungen überprüft und ausgeführt werden müssen, um das Ziel zu erreichen. Da sich das Ziel in der unteren rechten Ecke befindet, sollten Sie diese Richtung auch als erste Priorität verwenden. Dazu schlage ich folgende Methode vor: wenn (rechts) noch, wenn (rechts unten) noch wenn (rechte Hand-up) noch, wenn (links-unten) noch, wenn (links-oben) noch 0 umgeschrieben auf dem Funktionsschritt () es sieht aus wie: int step int xalt, int) printf getchar (); if(spielfeld[x][y] == 'O') { /* Sind wir am Ziel? */ spielfeld[x][y] = 'C'; spielfeld[xalt][yalt] = ' '; showspielfeld(); printf(Mister 'C' ist zu Hause!); exit (EXIT_SUCCESS); } else if(spielfeld[x][y]==' ') { spielfeld[x][y]='C'; spielfeld[xalt][yalt]=' '; showspielfeld(); /* rechts */ if(y+1 <49 &&= spielfeld[x][y+1]= !='*' &&= yalt!=y+1 &&= step(x,y+1,x,y)=)= return= 1;= rechts= unten= */= else= if(=></49> <49 &&=></49> <14 &&= spielfeld[x+1][y+1]= !='*' &&= xalt!=x+1 &&= yalt!=y+1 &&= step(x+1,y+1,x,y)=)= return= 1;= rechts= oben*/= else= if(= x-1=>0 && y+1 <49 &&= spielfeld[x-1][y+1]='*' &&= xalt!=x-1 &&= yalt!=y+1 &&= step(x-1 ,y+1,x,y))=return= 1;= nach= unten= */= else= if(=></49> <14 &&= spielfeld[x+1][y]= !='*' &&= xalt!=x+1 &&= step(x+1,y,x,y)=)= return= 1;= links= runter= */= else=></14> <14 &&= y-1=>0 && spielfeld[x+1][y-1]='*' && xalt!=x+1 && yalt!=y-1 && step(x+1,y-1,x,y)) rückgabe 1; /* nach oben */ else if(x-1>0 && spielfeld[x-1][y] !='*' && xalt!=x-1 && step(x-1,y,x,y)) return 1; /* nach links */ else if(y-1>0 && spielfeld[x][y-1] !='*' && yalt!=y-1 && step(x,y-1,x,y)) return 1; /* links oben */ else if(x-1>0 &&y-1>0 && spielfeld[x-1][y-1] !='*' && xalt!=x-1 && yalt!=y-1 && step(x-1,y-1,x,y)) return 1; spielfeld[x][y] = ' '; Rückgabe 0; Wenn Sie diese Funktion in ein vorheriges Beispiel einbeziehen, werden Sie feststellen, dass Herr C sie jetzt mit ein paar Hindernissen platzieren kann. Aber nach einer gewissen Anzahl von Hindernissen scheitert Mr. C wieder - und das ist, obwohl wir noch nicht in eine Sackgasse gekommen sind. Sie benötigen also noch eine Funktion, die sich daran erinnert, ob das Feld bereits besucht wurde oder nicht. Es ist eine einfache Aufgabe, nur mit einem anderen zweidimensionalen Array: alle Felder werden mit dem Wert 0 initiiert. Im Programm müssen Sie der bereits besuchten Position nur einen Wert von 1 hinzufügen. Dies bedeutet jedoch auch, dass Sie in einer Schrittfunktion () eine andere Bedingung in jeder Richtung überprüfen müssen. C: / <stdio.h> <stdlib.h></stdlib.h> </stdio.h> mister_c2.c</14> </14> </ENTER> #include #include #include . . #ifdef __unix__ #define clrscr () printf (»x1B»2J») #elif __BORLANDC__ »__MSDOS__ #include <conio.h>#elif __WIN32__ »_MSC_VER #define clrscr() (»cls») #else #define clrscr() printf(»clrscr() #endif #define HINDERNISSE 200 char spielfeld /» 1 ,0, 0, »int »15 »50 »; • createspielfeld int i, j, x, y; (i'0, j'0; j < 50;= j++)= spielfeld[i][j]='" ;= für(i=1 ;i=> < 15;= i++)= for(j=> <50;j++) = j==49) spielfeld[i][j]='" ;= spielfeld[i][j]=' ' ;= if(i==13 &&= j==48) spielfeld[i][j]='O' ;= für(i=></50;j++)> <50;j++) spielfeld[i][j]='" ;= für(i=></50;j++)> <=HINDERNISSE;i++) <> <50 &&= x=>0'y>0) spielfeld'x'y "; s spielfeld '1'1'; (i'0; i <15; i++)= for(j=0;></15;> <50; j++)= besucht[i][j]=0; =void= showspielfeld(void)= s= int= i,= j;= clrscr();= for(i=0; i=><<> < 15;= i++)= for(j=> <50;j++) '= printf(%c,spielfeld[i][j]);= if(j==49) printf();= '= '= int= step(int= x,= int= y,= int= xalt,= int= yalt)= '=></50;j++> < getchar (); if(spielfeld[x][y] == 'O') { /* Sind wir am Ziel? */ spielfeld[x][y] = 'C'; spielfeld[xalt][yalt] = ' '; showspielfeld(); printf(Mister 'C' ist zu Hause!); exit (EXIT_SUCCESS); } else if(spielfeld[x][y] == ' ') { besucht[x][y] = 1; spielfeld[x][y]='C'; spielfeld[xalt][yalt]=' '; showspielfeld(); /* rechts */ if(y+1 <49 &&= spielfeld[x][y+1]= !='*' &&= yalt!=y+1 &&besucht[x][y+1]!=1 &&= step(x,y+1,x,y))= return= 1;= rechts= unten= */= else= if(=></49> <14 &&=></49> <49 &&= spielfeld[x+1][y+1]= !='*' &&= xalt!=x+1 &&= yalt!=y+1 &&= besucht[x+1][y+1]!=1 &&= step(x+1 ,y+1,x,y))= return= 1;= rechts= oben= */= else= if(= x-1=>0 && y+1 <49 &&= spielfeld[x-1][y+1]!='*' &&= xalt!=x-1 &&= yalt!=y+1 &&= besucht[x-1][y+1]!=1 &&= step(x-1,y+1,x,y)=)= return= 1;= nach= unten= */= else= if(=></49> <14 &&= spielfeld[x+1][y]= !='*' &&= xalt!=x+1 &&= besucht[x+1][y]!=1 &&= step(x+1,y,x,y)=)= return= 1;= links= unten= */= else= if(=></14> <14 &&= y-1=>0 && spielfeld[x+1][y-1]!='*' && xalt!=x+1 && yalt!=y-1 && besucht[x+1][y-1]!=1 && step(x+1 ,y-1,x,y)) Rückgabe 1; /* nach oben */ else if(x-1>0 && spielfeld[x-1][y] !='*' && xalt!=x-1 && besucht[x-1][y]!=1 && step(x-1,y,x,y)) /* nach links */ else if(y-1>0 && spielfeld[x][y-1] !='*' && yalt!=y-1 && besucht[x][y-1]!=1 && step(x,y-1,x,y)) return 1; /* links oben */ else x-1 spielfeld'x-1'y-1' ! - Spielfeld (x'y';return 0; int main (Leerheit) createspielfeld (); Tonhöhe (1,1,1,1); Rückkehr EXIT_SUCCESS; Obwohl das ganze Thema ziemlich schwierig erscheinen mag, ist es ein logischer Prozess. Außerdem musst du dem Weg folgen, den du zu einem Ziel führen wolltest. nicht zurückkommen (wie wir wissen, führt es nicht zu einem Tor). Mit dieser Strategie gelangen Sie immer durch ein komplexes Labyrinth an Ihr Ziel (wenn es einen Weg zu Ihrem Ziel gibt und das Labyrinth nicht so kompliziert ist, dass es einen Überlaufstapel gibt).22.8.2 Das häufigere Beispiel für Rollback ist das 8-Lady-Problem. Die Herausforderung ist: Die Position der 8 Damen auf dem Schachbrett ist so, dass sie sich nicht selbst in Gefahr bringen. Für diejenigen, die es nicht wissen, kann eine Dame eine beliebige Anzahl von Feldern in der gleichen Spalte, Linie oder Diagonale von ihrer aktuellen Position bewegen - was bedeutet, dass keine andere Dame in diese Richtungen stehen kann. Versuchen Sie, es auf einem Schachbrett zu imitieren. Es gibt genau 92 Möglichkeiten. Sie können ein zweidimensionales Array für ein Schachbrett verwenden, aber da sich zwei Damen in einer Reihe oder Spalte sowieso gegenseitig bedrohen, können Sie es vermeiden. Da die First Lady, die Sie setzen, keine Angst bedroht, setzen wir sie direkt in die obere rechte Ecke. So könnte der Funktionsaufruf wie folgt aussehen: ein Schachbrett int; int i; für (i 0; i zlt; 8; i') Schachbrett (i) 0; / - Dame in der oberen linken Ecke - Dame (Schachbrett, 7); Zu diesem Zweck ein Teil der Funktion der Dame (): Int die Dame (int Schachbrett, int Position) - Int x Nr. 1; statischer Zähler Int Nr. 1; während (x'lt;'8) Der Wert x wird verwendet, um einzelne Damen zu identifizieren. Jede Dame bekommt eine Nummer. Darüber hinaus wird dieser Wert auch verwendet, um zu überprüfen, ob eine Dame diagonal bedroht ist. Aber mehr dazu später. Mit der Position der x Schachbrettlinie; bekommt Feld 7 (genaues Feld 8, aber da hier 0 auch als erstes Feld vorhanden ist, nur 0 bis 7 statt 1 bis 8) im oberen rechten Eckwert 1: Klicken Sie hier, um das Bild von Abbildung 22.40 zu vergrößern Die First Lady wurde in der oberen rechten Ecke platziert (1). Als nächstes benötigen Sie eine Funktion, um zu überprüfen, ob die Dame mit einer anderen Dame auf Platz 1 in einer Reihe, Spalte und Diagonale steht: wenn (!dame_in_gefahr)) die Funktion nur Daten vom Schachbrett überträgt. Im ersten Gang wird der Zustand logischerweise wahr sein - das heißt, die Dame ist nicht in Gefahr. Nun müssen Sie prüfen, ob Sie Position 0 nicht erreicht haben (im übertragenen Sinne wäre dies in der Spalte unten angekommen): Wenn Sie nicht die gesamte Spalte durchlaufen haben, beginnt der erste rekursive Anruf (und Stapelerhöhung) von hier aus: Noch einmal auf einen Blick: Int dame (int checkerboard, int position) - Int x q 1; statischer Zähler Int Nr. 1; Während (x zlt; x 8) wenn (!dame_in_gefahr (Checker Boards)) - wenn (Position) - / - die nächste Dame ... ... Put the lady in this position... . . . . . He said , he said that I. A- - I . . . . . Mit der Re-Call-Funktion sieht die Situation auf dem Schachbrett so aus: Klicken Sie hier, um Abbildung 22.41 Erste rekursive Anruffunktion zu vergrößern - eine weitere Dame Jetzt ist die Dame an Position 7 in Gefahr, und die folgende Bedingung gilt nicht: Wenn (!dame_in_gefahr (Schachbrett)) Also der Zähler x um 1: Visuell dargestellt erhöht, führt die nächste Codezeile nun zu folgendem Zustand auf dem Schachbrett: Klicken Sie hier, um das Bild zu vergrößern Abbildung 22.42 Ein weiterer Strich der Dame tritt auch an dieser zweiten Position auf. Also x wieder inkrementiert, so dass Sie den folgenden Zustand finden können: Klicken Sie hier, um auf Abbildung 22.43 Ein anderer Zug zu zoomen, und keine Kollision. Nun gefährden sich beide Damen nicht mehr gegenseitig, und so wird die rekursive Call-Funktion wieder ausgeführt. Der Zustand der aktuellen Funktion kehrt zum Stack zurück (zweite Funktion im Stapel) und wartet, bis sie erneut verwendet wird. Jetzt beginnt das Spiel von neuem. Der nächste Schritt ist folgender Schritt (siehe Abbildung 22.44). Klicken Sie hier, um Abbildung 22.44 Second Recursive Call Function Lady () zu vergrößern, da die Dame ohnehin vor den nächsten drei Reihen steht, verpassen wir diese drei Schritte, bei denen der x-Wert dreimal erhöht wird, bis die nächste Position erreicht ist: Klicken Sie hier, um Abbildung 22.45 zu vergrößern. Gut bemerkt, bedeutet dies nicht, dass dies die endgültige Position ist, da alle diese Feature-Aufrufe immer noch auf dem Stapel sind und darauf warten, was mit ihnen geschehen sollte (1 - bleibt so; 0 - Suche). Nach weiteren rekursiven Anruffunktionen passiert endlich etwas: Klicken Sie hier, um Abbildung 22.46 zu vergrößern. Der nächste Feature-Aufruf (für die erste Spalte) gibt nun den Wert auf 0 zurück, da keine Dame in der ersten Spalte platziert werden kann, ohne sie mit der anderen in Gefahr zu bringen. Verfolgen Sie nun unsere Funktion, bis die zweite Spalte wieder da ist. Das im Stapel gespeicherte x war 6 in der zweiten Spalte, so dass es wieder und setzt sich in Zeile 7 (zweite Spalte) fort. Es gibt eine weitere Kollision, wie in Zeile 8 (zweite Spalte). Somit erhält der Abonnent der zweiten Spalte auch 0 zurück, und unser Programm setzt seine Ausführung in der dritten Spalte (links) und der vierten Zeile (x 4 Wert hier) fort. Weitere Inkremente in dieser Spaltenspalte ne успех, а только дальнейшие столкновения. Это восходит до тех пор, пока либо нет столкновения (тогда он идет вперед снова) или условие становится неверным. Другими словами, мы достигли конца. Это подтверждается возвращением 1: если (позиция) если (дама (шашка, позиция-1)) возвращение 1; Поставить даму в такое положение... Мы сделали, у нас есть решение ... или если вы хотите потратить все 92 варианта. Мы использовали последнее. Рекомендую вам лучше понять всю программу, переиграть ее шаг за шагом на шахматной доске. Кроме того, полный список, которая решает проблему 8-леди программным путем: /... #include #include 8dame.c <stdio.h> <stdlib.h>dame_in_gefahr . <7; x++)= ist= auf= feld[x]= eine= dame?= */= if(schachbrett[x])= for(y=x+1;></7;> <=7; y++)= ist= auf= feld[y]= eine= dame?= */= if(schachbrett[y])= {= wir= überprüfen,= ob= die= beiden= *= damen= kollidieren.= */= sind= beide= damen= in= derselben= zeile?= */= if(schachbrett[x]==schachbrett[y]) return= 1;= kollision= in= gleicher= zeile= */= diagonal?= */= if(abs(x-y)==abs(schachbrett[x]-schachbrett[y])) return= 2;= kollision= in= der= diagonalen= */= }= return= 0;= keine= kollision!= */= }= int= dame(int= *schachbrett,= int= position)= {= int= x=1, i;= static= int= counter=1; while(x=></=7;> <= 8)= {= wir= setzen= die= dame= mit= der= *= nummer= x= an= feld[position].= */= schachbrett[position]=x; if(!dame_in_gefahr(schachbrett))= {= if(position)= {= die= nächste= dame= */= if(dame(schachbrett ,position-1))= return= 1;= dame= an= diese= position= setzen= */= }= else= {= printf(loesungs-nr.%2d= := ,= counter++);= for(i=0;></=> <8; i++)= printf((%d,%d),= i+1,= schachbrett[i]);= printf();= }= }= x++;= }= schachbrett[position]=0; return= 0;= }= int= main(void)= {= int= schachbrett[8],= x;= for(x=0; x=></8;> < 8; x++) schachbrett[x] = 0; dame(schachbrett,7); return EXIT_SUCCESS; } In der Zeile if(abs(x-y)==abs(schachbrett[x]-schachbrett[y])) wird eine absolute Zahl berechnet, das heißt beispielsweise, der absolute Wert von 2–6 ist 4 und nicht –4. Diese Berechnung und Bedingung dient dazu, zu überprüfen, ob in der Diagonalen eine Kollision mit einer Dame stattfindet. Hier sehen Sie noch eine der 92 möglichen Lösungen: Hier klicken, um das Bild zu vergrößern Abbildung 22.47 Keine Dame ist gefährdet. Ihre Meinung Wie hat Ihnen das Openbook gefallen? Wir freuen uns immer über Ihre Rückmeldung. Schreiben Sie uns gerne Ihr Feedback als E-Mail an kommunikation@rheinwerk-verlag.de. Page 14 CGI (Common Gateway Interface) ist, einfach ausgedrückt, eine Schnittstelle, mit der Sie z. B. Anwendungen für das Internet schreiben können. 23.1 Was ist CGI? Im Laufe dieses Kapitels werden Sie diese Frage detaillierter beantwortet bekommen, aber ein kurzer Anriss des Themas 8;= x++)= schachbrett[x]=0; dame(schachbrett,7);= return= exit_success;= }= in= zeile= if(abs(x-y)==abs(schachbrett[x]-schachbrett[y])) wird= eine= absolute= zahl= berechnet,= das= heißt= beispielsweise,= der= absolute= wert= von= 2–6= ist= 4= und= nicht= –4.= diese= berechnung= und= bedingung= dient= dazu,= zu= überprüfen,= ob= in= der= diagonalen= eine= kollision= mit= einer= dame= stattfindet.= hier= sehen= sie= noch= eine= der= 92= möglichen= lösungen:= hier= klicken,= um= das= bild= zu= vergrößern= abbildung= 22.47= keine= dame= ist= gefährdet.ihre= meinung= wie= hat= ihnen= das= openbook= gefallen?= wir= freuen= uns= immer= über= ihre= rückmeldung.= schreiben= sie= uns= gerne= ihr= feedback= als= e-mail= an= kommunikation@rheinwerk-verlag.de.= page= 14= cgi= (common= gateway= interface)= ist,= einfach= ausgedrückt,= eine= schnittstelle,= mit= der= sie= z.= b.= anwendungen= für= das= internet= schreiben= können.= 23.1= was= ist= cgi?= im= laufe= dieses= kapitels= werden= sie= diese= frage= detaillierter= beantwortet= bekommen ,= aber= ein= kurzer= anriss= des= themas=></ 8; x++) schachbrett[x] = 0; dame(schachbrett,7); return EXIT_SUCCESS; } In der Zeile if(abs(x-y)==abs(schachbrett[x]-schachbrett[y])) wird eine absolute Zahl berechnet, das heißt beispielsweise, der absolute Wert von 2–6 ist 4 und nicht –4. Diese Berechnung und Bedingung dient dazu, zu überprüfen, ob in der Diagonalen eine Kollision mit einer Dame stattfindet. Hier sehen Sie noch eine der 92 möglichen Lösungen: Hier klicken, um das Bild zu vergrößern Abbildung 22.47 Keine Dame ist gefährdet. Ihre Meinung Wie hat Ihnen das Openbook gefallen? Wir freuen uns immer über Ihre Rückmeldung. Schreiben Sie uns gerne Ihr Feedback als E-Mail an kommunikation@rheinwerk-verlag.de. Page 14 CGI (Common Gateway Interface) ist, einfach ausgedrückt, eine Schnittstelle, mit der Sie z. B. Anwendungen für das Internet schreiben können. 23.1 Was ist CGI? Im Laufe dieses Kapitels werden Sie diese Frage detaillierter beantwortet bekommen, aber ein kurzer Anriss des Themas > </stdlib.h> </stdio.h> . . . . hier bereits gegeben. Die Common Gateway Interface (CGI) ist eine Schnittstelle, die beispielsweise zum Schreiben von Anwendungen für das Internet verwendet werden kann. Diese CGI-Anwendungen werden auf einem (Web-)Server (z. B. Apache) ausgeführt und sind in der Regel über einen Webbrowser auf der HTML-Webseite verfügbar. Die Daten werden entweder von der HTML-Seite selbst (der POST-Methode) - z. B. über das Eingabeformular - oder direkt über die URL (GET-Methode) abgerufen. Die CGI-Schnittstellenprozedur ist ziemlich einfach. Die Daten werden in der Regel durch Standardeingabe (stdin) oder Umgebungsvariablen und Ausgabeausschnitte bei Bedarf ermittelt. Meistens handelt es sich um ein dynamisch generiertes HTML-Dokument, das Sie in Ihrem Browser anzeigen können. Sobald diese Voraussetzungen erfüllt sind, können CGI-Anwendungen mit fast jeder Programmiersprache erstellt werden. Das CGI-Programm, das Sie erstellen, ist ein Webserverprogramm, das nicht vom durchschnittlichen Benutzer, sondern von einem Webserver als neuer Prozess ausgeführt wird (siehe Abbildung 23.1). Was ist Ihre Meinung Wie gefällt Ihnen Openbook? Wir freuen uns immer auf Ihre Anhörung. Bitte senden Sie uns Ihr Feedback per E-Mail kommunikation@rheinwerk-verlag.de. Seite 15 Ist unbestreitbar, dass Perl dank seiner leistungsstarken Linienverarbeitungsfunktionen die Sprache ist, die am häufigsten zum Schreiben von CGI-Skripten verwendet wird. Perl (meiner Meinung nach) ist leichter zu erlernen als C, und der Aufwand, der mit der Programmierung verbunden ist, ist in Perl begrenzt. Wo man ganze Prozeduren in C schreiben muss, kann Perl nur mit ein oder zwei Zeilen auskommen. Es ist jedoch nicht der Zweck dieses Buches, die Vor- und Nachteile verschiedener Programmiersprachen zu diskutieren. Klicken Sie hier, um Abbildung 23.1 Stream in der CGI-Schnittstelle zu vergrößern Einige der Vorteile von CGI-Anwendungen, die in C erstellt wurden, sollten erwähnt werden: Es gibt weniger Zugriff auf den Server, auf dem sich das CGI-Skript befindet. Das bedeutet, dass Sie keinen Dolmetscher starten müssen, der das Skript übersetzen muss. C ist eine schnelle und systemorientierte Sprache. Somit ist die Lade- und Initialisierungszeit ausgeglichener als bei anderen Sprachen. Die Programme sind recht klein. Es gibt einen Compiler für alle Betriebssysteme. Da der Quellcode nicht zum Ausführen geöffnet ist, eignen sich die Sicherheitsfeatures hervorragend zum Ausblenden. Dieser Aspekt der Sicherheit muss hervorgehoben werden. Was ist Ihre Meinung Wie gefällt Ihnen Openbook? Wir freuen uns immer auf Ihre Anhörung. Bitte senden Sie uns Ihr Feedback per E-Mail kommunikation@rheinwerk-verlag.de. Seite 16 Der Hauptgrund für den ungezügelten Boom der Welt Das Web ist in erster Linie eine Plattform für die Unabhängigkeit der Server-Kunden-Beziehung. Es ist nicht erforderlich, auf das Betriebssystem, die Computerhardware oder die Netzwerkkonfiguration zu achten, solange Server und Client denselben Protokollstandard verwenden. Dank einzelner Protokolle (HTTP, FTP ...) hat die Verfügbarkeit verschiedener Webserver, Webbrowser, Programmiersprachen und Skripttechnologien dramatisch zugenommen. Webtechnologien sind in zwei Funktionale Kategorien (Server und Client) und drei konzeptionelle Ebenen (Ansichtsebene, Verarbeitungsebene, Speicherebene) unterteilt. Auf den nächsten Seiten werde ich näher darauf eingehen. Was ist Ihre Meinung Wie gefällt Ihnen Openbook? Wir freuen uns immer auf Ihre Anhörung. Bitte senden Sie uns Ihr Feedback per E-Mail kommunikation@rheinwerk-verlag.de. Seite 17 Die erste Stufe der Webprogrammierung ist die Präsentationsebene. Dies ist immer noch die häufigste Ebene. Der Webbrowser verwendet das HTTP-Protokoll, um eine Datei von einem Webserver anzufordern. Der Webserver sendet diese Datei dann an den Browser, was in der Regel die Darstellung einer einfachen Webseite ist. Die Präsentationsebene enthält das gesamte Erscheinungsbild der Seite, z. B. Klicks, Messwerte usw., nur alle Benutzerinteraktionen. Die Präsentationsebene wird mit Websprachen wie HTML oder JavaScript erstellt. Dies sind interpretierbare Sprachen, die nur analysiert werden, wenn eine Webseite Zugriff auf einen Webserver erhält. Hinweis Zum Anzeigen von Webseiten müssen Sie die Regeln befolgen, die der Webbrowser erwartet. Um die Webseitenzuordnung korrekt zu gestalten, sucht der Webbrowser nach einer bestimmten Struktur, die HTML enthält. Hypertext Markup Language (HTML) ist eine Seitenbeschreibungssprache, die zum Erstellen von Webdokumenten verwendet wird, die mit anderen Webdokumenten (Hyperlinks oder einfache Links) verknüpft sind. HTML ist keine Programmiersprache, sondern eine Reihe von Regeln, die eine Webseite formatiert. Beachten Sie, dass Sie als Web-Programmierer HTML verstehen müssen. Egal, ob Sie Ihre Webseiten in Skriptsprachen wie Perl oder PHP oder in C/C-Programmiersprachen schreiben, Sie kommen immer mit HTML in Kontakt. Obwohl ein Webprogrammierer viel Code für die nächste Verarbeitungs- und Speicherebene schreibt, wird es für Sie schwieriger sein, dynamische Dokumente an die Präsentationsebene zurückzugeben. Oft ist es auch möglich, dass die Anzeigeebene auch nur aus dynamischen Dokumenten besteht. Es muss also nicht immer eine statische Webseite sein. Da das Buch solche Bedingungen jedoch nicht voreinstellen kann, DIESE HTML-Ausgaben sind in Funktionen unterteilt. Damit Es Ihnen grundsätzlich egal ist, was diese Funktion verlässt. Alles, was Sie tun müssen, ist die richtigen Argumente an die Funktionen zu übergeben. Die Verarbeitungsebene führt Vorgänge wie Fehlerüberprüfung, Plausibilitätsprüfungen und Formularfelder durch. Wenn z. B. eine E-Mail von einer Webanwendung gesendet wird, wird auf der Verarbeitungsebene überprüft, ob die Eingabe korrekt ist. Es koordiniert dann, mit welcher E-Mail das E-Mail-Programm gesendet wird. Natürlich wird in der Verarbeitungsebene noch viel arbeitest. Mehr dazu erfahren Sie in diesem Kapitel. Die Verarbeitungsebene erfolgt oft mit interpretierbaren Skriptsprachen wie Perl, PHP, ASP und Programmiersprachen wie C/C. Interpretierte Sprachen sind bei bestimmten Verarbeitungsvorgängen recht langsam und funktionieren daher auf der Serverseite. Und um zum Beispiel mit dem E-Mail-Programm zurückzukehren: Auf der Serverseite ist garantiert, dass die E-Mail mit einem bestimmten Programm (in der Regel mit einem E-Mail-Programm) gesendet wird. Auf der Seite des Kunden wiederum erfordert dies, dass der Benutzer dieses Programm auf seinem Computer installiert. Es ist nicht unbedingt bequem.23.4.3 Speicherschicht Irgendwann möchten Sie auch Daten speichern. Es gibt viele Gründe für das Speichern von Daten, sei es Statistiken, Berechnungen oder Registrierungsdaten. Und je härter und umfangreicher die Daten, die Sie speichern werden, desto besser sollte die Lösung sein. Eine Textdatei ist für einen einfachen Besucherzähler ausreichend, da sie jeweils nur eine Zahl hat. Aber wenn Sie Apps wie einen Online-Shop erstellen möchten, können Sie die Datenbank nicht mehr umgehen. Wenn Sie Daten ändern, löschen oder Teile der Daten extrahieren müssen, geschieht dies häufig mit etwas Code in der Datenbank. Heute stehen auch viele Datenbanklösungen für herstellerspezifische Erweiterungen zur Verfügung. Und am wichtigsten ist, dass viele dieser Lösungen wenig Geld kosten. Mehr dazu erfahren Sie in Kapitel 24, MySL und C. Ihre Meinung gefällt Ihnen Openbook? Wir freuen uns immer auf Ihre Anhörung. Bitte senden Sie uns Ihr Feedback per E-Mail kommunikation@rheinwerk-verlag.de. Seite 18 stehen dem Webprogrammierer auf der Clientseite derzeit zahlreiche Programmiermethoden zur Verfügung. Einige davon hängen von Systemen ab und sind daher weniger nützlich für die Webprogrammierung. Hier ist ein kurzer Überblick über die Client-Sprachen von Skripten und deren Vor- und Nachteile.23.5.1 JavaScript JavaScript ist eine Teilmenge von Java, die direkt im Webbrowser implementiert wird. Mit JavaScript können Sie Prüfen, berechnen, öffnen Sie die Fenster für die Messung und vieles mehr. JavaScript: Wenn das Skript in einem Browser funktioniert, kann ein anderer dazu führen, dass die Webseite nicht einmal angezeigt wird. Und noch ein Nachteil: Der Surfer kann JavaScript komplett ausschalten. Wenn Sie die gesamte javaScript-Navigation erstellen, war die Arbeit vergeblich. Und sie werden wahrscheinlich nie wieder einen Surfer sehen. Wenn Sie also JavaScript für Ihre Website auswählen, vergessen Sie nie, es zu verwenden. Dies bedeutet für Webbrowser, die JavaScript nicht verstehen, dass der Code zwischen diesen Tags funktioniert. JavaScript wurde von vielen verschiedenen Variationen von Microsoft begleitet, z. B. Jscript und VBScript, wobei Jscript eine Fortsetzung von JavaScript und VBScript basierend auf der Visual Basic-Syntax ist. Wenn Sie jedoch eine dieser beiden Skriptsprachen verwenden möchten, schließen Sie Surfer mit anderen Systemen aus.23.5.2 Java-Applets haben nichts mit JavaScript zu tun, außer der Ähnlichkeit des Namens. Java ist eine komplette Programmiersprache, eine Evolution zu C, die von SUN veröffentlicht wurde. Auf diese Weise können Sie komplexe Menüs für interaktive Spiele im Internet erstellen. Der Vorteil von Java-Applets ist die Unabhängigkeit der Plattform und Sicherheit, da der Code nicht sichtbar ist. Was ist Ihre Meinung Wie gefällt Ihnen Openbook? Wir freuen uns immer auf Ihre Anhörung. Bitte senden Sie uns Ihr Feedback per E-Mail kommunikation@rheinwerk-verlag.de. Seite 19 Heute gibt es viele Möglichkeiten, Serveranwendungen zu programmieren. Es gibt derzeit unzählige Schnittstellen, die dynamische Webseiten generieren und an den Webserver senden. Es gibt zwei Arten der Webprogrammierung auf der Serverseite: Eine basiert auf der Objektkompilierung und die andere verwendet ein Skriptsprachenmodell. Der Unterschied zwischen den beiden Arten ist einfach, aber immer noch klar. Szenariosprachen werden auf einem Webserver von einem Übersetzer während der Zeit der Zeit der Zeit der Strukturierung analysiert. Die gesammelten Objekte können auf dem Webserver ausgeführt werden und müssen nur gestartet werden. Daher sind Skriptsprachen in der Regel etwas langsamer als der kompilierte Code. Der Zweck der Serverprogrammierung ist jedoch immer derselbe - der dynamische HTML-Code sollte die Ausgabe als Webseite in Ihrem Browser sein. Nun muss die CGI-Schnittstelle besprochen werden. CGI wird oft als Programmiersprache bezeichnet, ist es aber nicht. CGI ist ein standardisiertes Gateway, das eine Schnittstelle zwischen einem Webserver und Code auf der Serverseite bereitstellt. Die Tatsache, dass CGI immer noch so beliebt ist wegen seiner Mehrsprachigkeit. Sie können CGI-Skripte mit jeder Programmiersprache zu schreiben, die etwas ausarbeiten kann. Der einzige Nachteil von CGI ist die Verarbeitung des Skripts, da jede CGI-Anforderung einen separaten Prozess in einem separaten Speicherbereich auf dem Webserver erfordert. Die Entwickler von Webservern ließen diesen Fehler jedoch nicht auf sich allein gestellt werden, und sie arbeiteten hart daran, das Problem zu lösen. Entwickler haben APIs für Webserver (App-Programmierschnittstellen) veröffentlicht. Diese APIs wurden vollständig in den Webserver integriert und arbeiteten tendenziell schneller als einfache Skripts. Der Nachteil solcher APIs ist jedoch, dass sie mit einem Webserver verschweißt sind. Dies ist nur für Webserver sinnvoll, die auf mehreren Systemen (z. B. Apache) verfügbar sind. Zu diesen eigenständigen Skriptsprachen gehören ASP (Active Server Pages), JSP (Java Server Pages), CFML (Cold Fusion Markup Language) und letztlich die immer beliebter werdende (oder bereits beliebte) phP-Skriptsprache. Aber es gibt auch eigenständige CGI-Anwendungen für Perl, wie mod_perl oder FastCGI. Wie gefällt Ihnen Openbook? Wir freuen uns immer auf Ihre Anhörung. Bitte senden Sie uns Ihr Feedback per E-Mail kommunikation@rheinwerk-verlag.de. Seite 20 des Internets, oder besser gesagt das World Wide Web, wurde nach dem Prinzip des Clients/Servers entwickelt. Ein Webbrowser ist ein Kunde in dieser Architektur. Wenn Sie beispielsweise Adresse in einen Webbrowser eingeben, ruft der Browser Dienste online ab. In diesem Fall wird die HTTP-Anforderung über das Netzwerk gesendet. Mit dem HTTP-Protokoll haben Sie nun Zugriff auf Dateien von Servern in den Netzwerken. Der Server, genauer gesagt der Webserver, verarbeitet derzeit die Anfrage des Kunden. Der Standort des Webservers spielt keine Rolle. Es könnte sich im selben Gebäude, auf der anderen Seite der Erde oder auf demselben Computer befinden. In den meisten Fällen sendet der Webserver des Webservers das angeforderte Dokument an den Kunden, z. B. HTML-Daten, um eine Webseite zu präsentieren. Ein Webserver wird oft als Hypertext-Übertragungsprotokoll-Dämon (HTTP) bezeichnet. Abbildung 23.2 veranschaulicht den Prozess einer solchen Transaktion: Klicken Sie hier, um Abbildung 23.2 Transaction http request zu vergrößern Hinweis Webserver ist nichts anderes als ein Programm oder eine Anwendung (in der Regel in C geschrieben) ohne visuelle Schnittstelle auf dem Server. Sie können das Programm nicht sehen. Es dauert Arbeit durch Ports (in der Regel Port 80) und führt es aus, in der Regel eine HTML-Seite anzeigen. Oft ist der Server als Hardware fälschlicherweise gleichbedeutend mit einem Serverprogramm, aber die Beziehung hier ist so nah, dass es auch völlig falsch. 23.7.2 Durch die Implementierung eines Servers vor der Auswahl einer Online-Präsenz, ob privat oder für eine Organisation, müssen Sie zunächst überlegen, wie oder mit wem Sie online gehen. Die Kosten und der Zweck Ihrer Anwesenheit sind wahrscheinlich entscheidend für Ihre Organisationsstrategie. In der Regel wählen Sie eine der beiden häufigsten Webserverimplementierungsstrategien: Hostingdienst - Der Webserver wird remote über einen IsP unterstützt. Schlüsselfertige Lösungen – Dies ist eine komplette Software- und Harware-Lösung, die lokal auf einer physischen Site verwaltet wird. 23.7.3 Hosting-Dienste Die meisten Nutzer entscheiden sich eher für Hosting. Der Kunde gelangt über einen Webprovider (WSP) indas Netzwerk. Verwech

Sie können Erweiterungsänderungen speichern, wenn Sie diese Erweiterung direkt von der Übersetzungszeit an die Ausführung der Datei übergeben: gcc-o time.cgi.c. Fehlerbehebung Wenn Sie keine CGI-App erhalten und ausgeführt werden, überprüfen Sie die folgende Checkliste, um zu sehen, was die mögliche Ursache sein könnte: Haben Sie die Zugriffsrechte für die CGI-App korrekt eingerichtet? Wenn Sie nicht über die Rechte zur Datei verfügen, sollten Sie dies tun. Gibt es einen Fehler im Programm? Meistens gibt Ihr Webbrowser den Fehler mit der Nummer 500 zurück. Starten Sie das Programm in der Befehlszeile und achten Sie auf den Ausgang. Haben Sie die richtige URL in den Browser eingegeben? Überprüfen Sie erneut, ob der Webserver ordnungsgemäß oder überhaupt funktioniert (. Ein Blick auf das Apache-Magazin kann auch nicht schaden. Dies ist in der Regel in /var/log/apache/ der Fall. Die Dateien error.log und access.log sind hier interessant. Insbesondere beim Testen neuer CGIs sollten Sie tail-f/var/log/apache/error.log ausführen, indem Sie die Konsole eingeben. Alle Fehlermeldungen, die dann über stderr ausgegeben werden, werden im Protokoll angezeigt. Bevor Sie weitermachen Sie benötigen noch einige Grundkenntnisse über das http-Protokoll. Ihre Meinung Wie gefällt Ihnen Openbook? Wir freuen uns immer auf Ihre Anhörung. Bitte senden Sie uns Ihr Feedback per E-Mail kommunikation@rheinwerk-verlag.de. Seite 21 Hypertext Protocol (HTTP) ist ein Protokoll, das von einem Webbrowser und web server verwendet wird, um sich gegenseitig über das Internet zu behandeln. Und da CGI auf dem HTTP-Protokoll basiert, ist es sinnvoll, es ein wenig vertraut zu machen. Natürlich ist es nicht meine Absicht, Ihnen alles zu diesem Thema zu sagen, sondern nur das Wichtigste. Wenn Sie weitere Informationen benötigen, kann ich eine Webseite www.w3.org/Protocols/.23.8.1 Webprotokollen empfehlen Der Zweck dieses Abschnitts besteht darin, zu erklären, wo das HTTP-Protokoll seinen Platz in der Netzwerktechnologie hat. Natürlich gibt es weit mehr Protokolle für das Internet als nur HTTP. Beispiele hierfür sind das File Transfer Protocol (besser bekannt als FTP), das TELNET-Protokoll, das Simple Mail Protocol (SMTP) und vieles mehr. Für Sie als CGI-Programmierer ist das HTTP-Protokoll von Interesse.23.8.2 Wozu sind die Protokolle? Grundsätzlich können Sie Netzwerktechnologieprotokolle mit Sprachen vergleichen. Menschen auf der ganzen Welt sprechen Tausende von verschiedenen Sprachen. Wenn du mit jedem auf der Welt sprechen willst, musst du all diese Sprachen lernen - eine schwierige Sache. Wenn das gleiche auf Computer angewendet wurde, wäre es schwierig, die Website zu sehen. Sie erhalten wahrscheinlich eine Fehlermeldung wie: Diese Webseite kann nicht angezeigt werden, weil Ihr System das Protokoll nicht versteht.23.8.3 Was ist das Protokoll? Protokolle sind eine Mischung aus Kommunikationsregeln und Nachrichtenformaten, denen Netzwerkcomputer folgen müssen, wenn Sie Daten miteinander teilen möchten. Wenn beispielsweise Daten von Ernies Computer an Berts Computer gesendet werden, werden sie zuerst in eine Nachricht gepackt. Diese Meldung wird in einer bestimmten Protokollkommunikationsregel definiert. Die Kommunikationsregel beschreibt beispielsweise, wie eine Nachricht auf dem Ziel-Bert-Computer im Falle eines Übertragungsfehlers verarbeitet wird. Das Internet Protocol Protocol (IP) wird zur Datenübertragung verwendet. Die Sprache der Computer ist nicht so komplex wie die Sprache der Menschen. Der Computer kommuniziert mit anderen Gleichgesinnten über die Binärzahlen 0 und 1. Trotz dieser universellen Sprache sind nicht alle Computer vom gleichen Typ. Von unterschiedlicher Hardware zu einem anderen Betriebssystem kann es ein ziemliches Kommunikationsproblem geben. Und Protokolle sind sozusagen Übersetzer, die Kommunikationsprobleme lösen. Das macht es nur dass verschiedene Computertypen mit verschiedenen Betriebssystemen kommunizieren können.23.8.4 Standards für Netzwerktechnologien, so dass Netzwerkcomputer nun über einzelne Protokolle miteinander kommunizieren können, musste zunächst eine einzige Netzwerktechnologie entwickelt werden. Auch die Standardisierung ist nach und nach erforderlich, da die schrittweise Entwicklung von Netzwerkkomponenten im Laufe der Zeit zu immer mehr Fortschritten in herstellerspezifischen Netzwerken und Protokollen geführt hat. Offensichtlich sind mit der Zunahme der Netze immer mehr Probleme aufgetreten. 1978 traf sich ein Gremium namens ISO (International Organization for Standardization), um eine Schnittstelle zu standardisieren, die für Netzwerk-PCs in Betracht gezogen werden könnte. Es dauerte dann noch sechs Jahre, bis das ISO-Komitee die Norm auserklärte. Der Standard wurde Open System Interconnection (OSI) genannt. Mit OSI wurde bestimmt, wie verschiedene Computer mit verschiedenen Protokollen kommunizieren sollen.23.8.5 Das OSI-Schichtmodell Mit der Einführung des OSI-Modells hat das OSI-Modell endlich einzelne Produkte für den Netzwerkeinsatz entwickelt, da die Hersteller gezwungen waren, sich auf diese Spezifikationen zu konzentrieren. Es sollte jedoch hinzugefügt werden, dass dies an sich nicht die Umsetzung aller Standards bedeutet. Das OSI-Benchmark-Modell ist ein bewährtes und bewährtes Modell für die Hard- und Softwareentwicklung. Ob und wie die Hersteller dieses Modell anwenden, bleibt abzuwarten. Das Layer-Standardmodell wurde auf sieben Ebenen verteilt. Jede Schicht hat ihre eigene Aufgabe. Abbildung 23.9 bietet einen kurzen Überblick über diese sieben Ebenen und Protokolle, die jede Ebene verwenden kann. Klicken Sie hier, um das Bild von Abbildung 23.9 des OSI-Layer-Modells zu vergrößern, und die hier nicht gezeigten Protokolle sind UDP-Magazine. Das HTTP-Protokoll (in der Abbildung grau dargestellt), das für die CGI-Programmierung verwendet wird, befindet sich über dem TCP-Protokoll. Der gesamte Webdatenverkehr durchläuft TCP/IP-Protokoll.23.8.6 Elemente der URL Oft stoßen Sie auf die URL (Single Locator-Ressource), deren Bedeutung auf den ersten Blick fremd erscheint. Betrachten Sie die folgende URL: diese URL muss jetzt in Teile davon unterteilt werden. Http ist ein Protokoll, das von einem Webbrowser verwendet wird, um mit einem Webserver zu kommunizieren. Natürlich gibt es neben HTTP noch weitere Protokolle. Mehr dazu erfahren Sie direkt bei den Ports. Darauf folgt der Name des Computers. www.pronix.de (Computername) Www.pronix.de ist der Name des Computers, auf dem die Website oder Ist. Es kann sich jedoch auch um eine IP-Adresse (z. B.) erfreuen und muss nicht immer ein Domänenname sein ( ). 80 (Portnummer) Webserver akzeptiert hier über Port 80 die Bestellung. Die Tatsache, dass dieser Port nicht auf der URL aufgeführt werden sollte, da Port 80 der Standardport des HTTP-Protokolls ist. Daten sollten jedoch nicht immer über Port 80 übertragen werden. Sie können auch einen Server einrichten, um Daten über Port 85 abzurufen. Die folgenden Ports und ihre Protokolle sind in der Regel festgelegt: Tabelle 23.3 Allgemeine Protokolle und ihre Ports Protokoll-Port-FTP (Datenverbindung) 20 FTP (Control Connection) 21 Telnet 23 SMTP 25 DNS 53 http 80 HTTPS (SSL-verschlüsselt) 444 3 POP3 110 NNTP 119 LDAP (Standard) 389 LDAP (SSL-verschlüsselt) 636/cgi/name.cgi (Pfadspezifikation) Dies gibt den Speicherort des gespeicherten Dokuments auf dem Webserver an. Die Abfragezeile kann aus mehreren Paaren von Namen und verwandten Werten bestehen. In diesem Beispiel haben Sie zwei Paare: die erste Paarvariable: den Namen; Bedeutung des ersten Paares: Clint-Variable des zweiten Paares: Nachname; Wert des zweiten Paares: East Line Request wird noch ausführlicher behandelt.23.8.7 Kundenanfrage - HTTP Request (Browser Request) Das erste, was beim Verbinden von HTTP ist immer eine Anforderung (Anforderung HTTP) Web-Browser an den Web-Server. Wenn Sie beispielsweise die URL in Ihrem Browser eingeben und darauf klicken, sendet der Browser eine Anforderung an einen Webserver, der aus zwei Teilen besteht: einem aus der Abfragezeile und dem anderen aus dem Kopfzeilenfeld. Es sieht so aus: GET/index.html HTTP/1.1 Host: www.pronix.de Akzeptieren: image/gif, image/wordpress, image/p-word, MSIE 6.0; Windows NT 5.1) Referenz: ... ... Die erste Zeile ist die so genannte Abfragezeile: GET ist eine Anforderungsmethode, mit der eine Ressource von einem Server angefordert wird, ohne die Daten auf dem Server zu ändern. GET ist die Standardmethode, wenn der Browser nach einem Dokument auf HTTP fragt. In diesem Beispiel wird nach einer datei index.html gefragt. Das Protokoll, über das der Webbrowser und webserver http mit der Versionsnummer 1.1 sprechen soll. Eine weitere häufig verwendete Abfragemethode ist POST. Im Wesentlichen unterscheiden sich diese beiden Methoden der Darstellung von Daten in der Art und Weise, wie Daten an einen Webserver übertragen werden. Um den Unterschied zwischen den beiden METHODS der Anforderung von GET und POST ein wenig klarer zu machen, hier ist eine schnelle theoretische GET Hier sollte die Eingabe des Textfensters für das HTML-Formular ausgewertet werden. Um dies zu tun, verwende ich den folgenden HTML:qlt;form action' method'get'lt;lt;b'gt;Bitte geben Sie Ihren Namen ein: qlt;b'gt;'lt;input value'hallo name'Textfeld size'20'gt; Dadurch wird das folgende Textfeld mit einer Schaltfläche erstellt: Klicken Sie hier, um Abbildung 23.10 zu vergrößern Das Eingangsfeld mit der C-Linien-Schaltfläche ist eine qlt;form action/ method'get'gt; das CGI auswert.cgi-Programm mit der GET-Methode anfordern. Sie können auch sehen, wie Sie die http-Anforderungsmethode bestimmen. Wenn Sie eine Taste auf dem Server drücken, wird die CGI-App aufgerufen. Ein Blick auf die URL des Webbrowsers zeigt, dass Sie jetzt die Abfragezeile auf dem Webserver mit der CGI Auswert.cgi App aus einer Variablenumgebung gleichen Namens QUERY_STRING lesen können. In dieser Variable hat nun die folgende Zeile: QUERY_STRING Textfeldhallo POST Wenn Sie dasselbe mit der POST-Methode tun möchten, müssen Sie nur Folgendes in HTML ändern: zlt;form action/ method/post'gt; jetzt gibt es keine Abfrage mehr in der URL auf Knopfdruck. In diesem Beispiel müssen Sie eine Variablenumgebung anfordern, CONTENT_LENGTH die Anzahl der Bytes, die diese Nachricht enthält. Dann können Sie CONTENT_LENGTH aus der Standardeingabe (stdin) lesen - z.B. mit Fgets (): Fgets (Puffer, CONTENT_LENGTH, Stdin); Danach befindet sich der vollständige Eintrag (Textfeld) des HTML-Formulars in der Pufferzeile. Sowohl die GET-Methode als auch die POST-Methode codieren die Daten und müssen noch entschlüsselt werden. Sie werden bald lernen, wie man es macht. Andere Methoden gibt es ein paar andere Methoden, aber ich gebe sie hier nicht ein. Diese Methoden sind praktisch ungenutzt. Hier ist ein kurzer Überblick über andere mögliche Abfragen, die Sie auf einem Webserver stellen könnten: HEAD - empfängt nur und keine Inhalte vom Webserver. PUT erstellt eine Ressource auf dem Server. DELETE - Entfernt die Ressource auf dem Server. OPTIONS wird verwendet, um mögliche Methoden zum Anfordern eines Servers anzufordern. Eine Überschrift anfordern Zusätzlich zur Abfragezeile sendet der Browser dem Webserver einige Informationen, den so genannten Abfrageheader. Diese Informationen bestehen aus dem Namen des Feldes, das durch den Doppelpunkt von seinem Wert getrennt ist. Nochmals das Beispiel der HTTP-Abfrage, die Sie bereits oben gesehen haben: GET/index.html HTTP/1.1 Host: www.pronix.de Akzeptieren: image/gif, image/word of words, image/psrc, zlt;/form/gt;'lt;'gt;de Connection: Keep-Alive User-Agent: Mozilla/4.0 (kompatibel; MSIE 6.0; Windows NT 5.1) Referenz: ... In diesem Beispiel ist die fett formatierte Schriftart der Titel der Abfrage. Cm. Tabelle 23.4 zur Überprüfung einiger . Sie können diese Informationen fast immer senden, wenn Sie einen Webbrowser verwenden und eine neue Webseite von einem Webserver anfordern. Tabelle 23.4 Einige Abfrageüberschriften und ihr Abfrageüberschriftswert Bedeutet Host ist der Zielcomputer, auf dem das Dokument angefordert wird. Dieser Titel wird verwendet, um sicherzustellen, dass der Server weiß, wie viele Daten aus dem Text der HTTP-Anforderungsnachricht gelesen werden sollen. Dies gilt natürlich nur für die POST-Anforderung (siehe POST). Dieses Feld ist für die GET-Anforderung nicht angegeben. Wenn die Anforderung über einen Nachrichtentext verfügt, sollte der Titel des Inhaltstyps zusammen gesendet werden. Dadurch wird der Typ des Medientexts bestimmt. Es enthält Informationen darüber, welcher Webbrowser und welches Betriebssystem die Anfrage gestellt hat - obwohl diese Informationen nicht immer korrekt sein müssen. Viele Programme und Webbrowser bieten die Möglichkeit, diesen Titel zu manipulieren. Sie enthält in der Regel die URL, die der Benutzer zuletzt besucht hat. Wenn ein Benutzer die URL jedoch direkt in einen Webbrowser eingibt oder ein Lesezeichen verwendet, ist in diesem Titel nichts vorhanden. Es gibt mehrere Empfangsrohlinge. Darin sendet der Browser Serverdaten, die die Arten von Antworten (Antworten vom Server) verstehen können. Beispiel: Akzeptieren: Eine Liste der Medientypen, die der Browser (Kunde) akzeptiert. Accept-Charset: Eine Liste von Zeichensätzen, die der Browser akzeptiert. Accept-Language: Die Sprachen, die der Browser annimmt. So informiert der Browser den Server, ob er Cookies akzeptiert. Wenn Sie beispielsweise Cookies in Ihrem Browser deaktivieren, wird dieser Titel nicht zusammen gesendet. Wenn Cookies akzeptiert werden, kann der Server ein Cookie installieren. 23.8.8 Serverantwort Nach sorgfältiger Verarbeitung der HTTP-Anforderung von einem Webbrowser antwortet der Webserver auch irgendwann mit einer HTTP-Antwort. Die Antwort besteht aus einer Statuszeile sowie einigen Kopffeldern und häufig dem Text der Nachricht. Hier ist ein Beispiel für eine Webserver-Antwort: HTTP/1.1 200 OK Datum: St. 30. Oktober 2002 01:21:22 GMT Server: Apache/1.3.14 Letzte Änderung: W, 29. Oktober 2002 22:21:19 GMT Inhaltslänge: 2232 Content-Type: text html q-lt;gt; Der Statuscode in der ersten Zeile, der Statuszeile, definiert zusätzlich zum Protokoll mit der Versionsnummer einen dreistelligen Statuscode. Dieser Statuscode gibt dem Webbrowser eine Antwort auf seine Anforderung. In diesem Beispiel wird der Statuscode auf 200 zurückgeführt, was zu einer erfolgreichen Abfrage führt. Ziemlich oft haben Sie den Statuscode 404 im Internet kennengelernt. Diese wird zurückerstattet, wenn das angeforderte Dokument nicht gefunden wird. Es folgt eine Übersicht über mögliche Statuscodes. In Tabelle 23.5 sind fünf Codegruppen aufgeführt, die sich von der ersten Ziffer unterscheiden. Tabelle 23.5 Statuscodes Einzelner Bereiche Statusbereich Wert 100-199 Gesamtinformationen. Die Anforderung wird noch bearbeitet. 200-299 Gut. Erfolgreiche Anfrage 300-399 Statusänderung, Neuansiedlung (oder Umleitung), weitere Maßnahmen müssen ergriffen werden, um eine erfolgreiche Anforderung zu gewährleisten. 400-499 Fehlermeldung vom Kunden (meist Webbrowser) 500-599 Fehlermeldung von Server Tabelle 23.6 zeigt mehr gemeinsame Statuscodes und ihre Werte: Tabelle 23.6 Allgemeiner Statuscodes und ihr Wertstatuscode bedeutet 200 alle gut 201 POST-Team erfolgreich angenommen 202 Anfrage 203 GET Anforderung durchgeführt 204 Anforderung durchgeführt 204 Anforderung erfüllt 204 Anforderung durchgeführt 204 Anforderung durchgeführt 204 Anforderung durchgeführ

>-€) (Größe (CGI_DATEN)); (ptr_anfang->-NULL) ptr_daten - ptr_anfang-> j'; - res - strtok () ; ptr_daten->- malloc (strlen (strlen(res);1 ptr_daten); ptr_daten->-res; res s strtok(null, -0); ptr_daten->value - malloc (strlen(res) 1 ptr_daten ptr_daten ptr_daten ptr_daten); • ptr_daten->-€) (CGI_DATEN)); (ptr_daten->-q NULL) ptr_daten - ptr_daten-> j'; ptr_daten; /* CGI_DATEN / ptr_anfang; Использовать для создания функции, которая готова для всех случаев. Теперь все, что вам нужно, это функция, которая освобождает динамическое пространство списка. Das Openbook ist ein Openbook? Мы всегда с нетерпением ждем ваших отзывов. kommunikation@rheinwerk-verlag.de . . . . . Страница 25 Чтобы продемонстрировать все функции сейчас, создать простую гостевую книгу. Думаю, все это знают. . . . . В этом примере HTML-файл, отображаемый записи, также использовался для хранения данных, чтобы сохранить программу немного короче. Для обширной гостевой книги для этой цели будет доступен дополнительный файл или база данных.23.12.1 Форма HTML (guestbook.html) <html><head><title>Вход в гостевую книгу</title></head><body text=#000000 bgcolor=#FFFFFF link=#FF0000 alink=#FF0000 vlink=#FF0000><h3>Форма для входа в гостевую книгу</h3><pre><form action= method=post><input value=IhrName name=Name size=20><input value=Mailadresse@mail name=E-Mail size=20><select name=Bewertung size=3><option>Примечание 1<option>Примечание 2<option selected=>Примечание 3<option>Примечание 4<option>Примечание 5<option>Примечание 6</select>
Ваша запись :<textarea name=textform cols=32 rows=6></textarea><input type=checkbox name=programmieren value=C/C++><input type=checkbox name=programmieren value=Perl<b<input type=Perl&lt gt;<input type=checkbox name=programmieren value=Visual Basic><input type=reset><input type=submit value=Abschicken></form><<<<</body></html> . . . . . CGI: <form action= method=post> 23.21 HTML form23.12.2 - CGI (»evade.cgi») / evade.c / #include <stdio.h> #include <stdlib.h> #include #define MAX_PAARE <string.h> 255 void print_location (char); char getdata-;); char 'Strdup(');; Hex2ascii () () char a (; CGI_DATEN ( ) printf_error () struct CGI_DATEN char s; char s; CGI_DATEN ; </string.h></stdlib.h></stdio.h></form>CGI_DATEN - NULL; URL; URL-URL- URL, ... -/ n print_location (char s url) printf (:: %s, url); /) , ...(Inhaltstyp: text/html); printf (<html><head>); printf (<title>'%s</title"; gt;url); printf (</head><body>); printf ('<a href='%s'>',</a</a</a</a</a<a<a<a</a<a</a<a<a</a</a</a</a</a</a</a</a</a</a</a<a</a<a gt;url); printf (</body></html>); . . . . . * Rückgabewert: String-Puffer mit den Daten * im Fehlerfall : NULL */ char *getdata(void) - unsignierte lange Größe; char *buffer = null; char *request = getenv(REQUEST_METHOD); char *cont_len; char *cgi_string; /* überprüfen Sie zuerst die Anforderungsmethode */ wenn(NULL == Anforderung) NULL zurückgibt; sonst wenn(strcmp(request, GET) == 0) . . . . . cgi_string QUERY_STRING cgi_string > . . . . . cgi_string . . . . . ./ /* Return-Wert an den Aufrufer */ ' andernfalls, wenn (strcmp(request, POST) == 0) - /* die Methode POST -> Länge der Zeichenfolge * (CONTENT_LENGTH) */ cont_len = getenv(CONTENT_LENGTH; finden; if(null == cont_len) geben NULL zurück; oder..... CONTENT_LENGTH . . . . . toi(cont_len); if(size <= 0)= return= null;= keine= eingabe!?!? = */= jetzt= lesen= wir= die= daten= von= stdin= ein= */= puffer=(char *)= malloc(size+1);= if(= null== puffer=)= return= null;= if(= null== fgets(puffer,= size+1,= stdin)= = den= aufrufer= */= return= puffer;= '= '= '