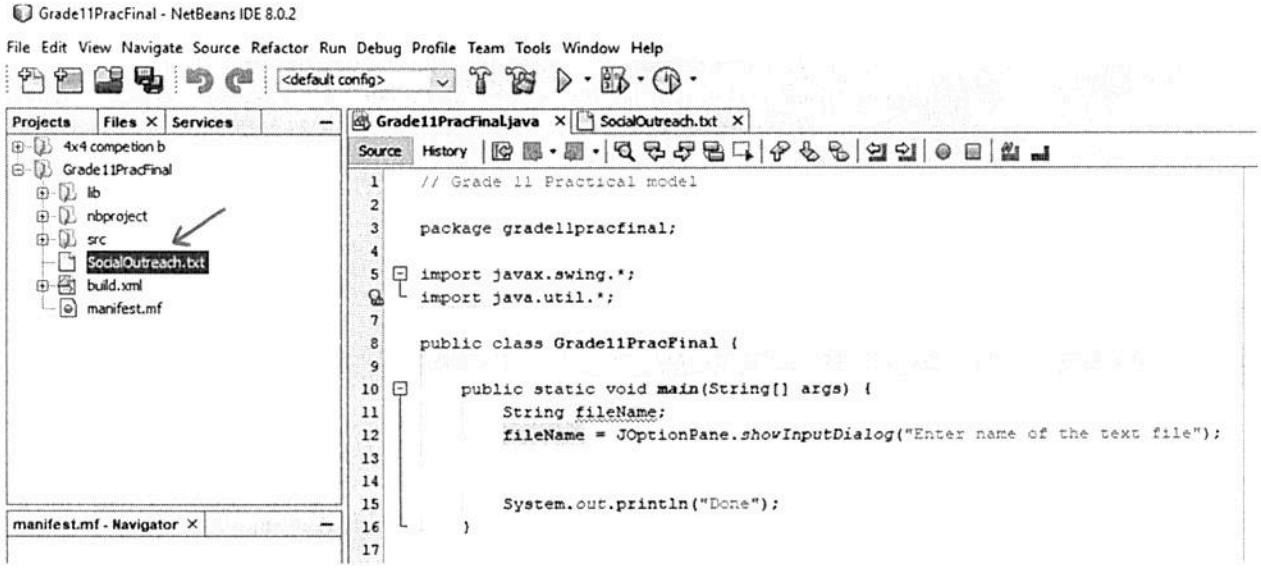
**Grade 11. Reading from a text file into an array and NetBeans.**

**A one class solution for teaching purposes only.**

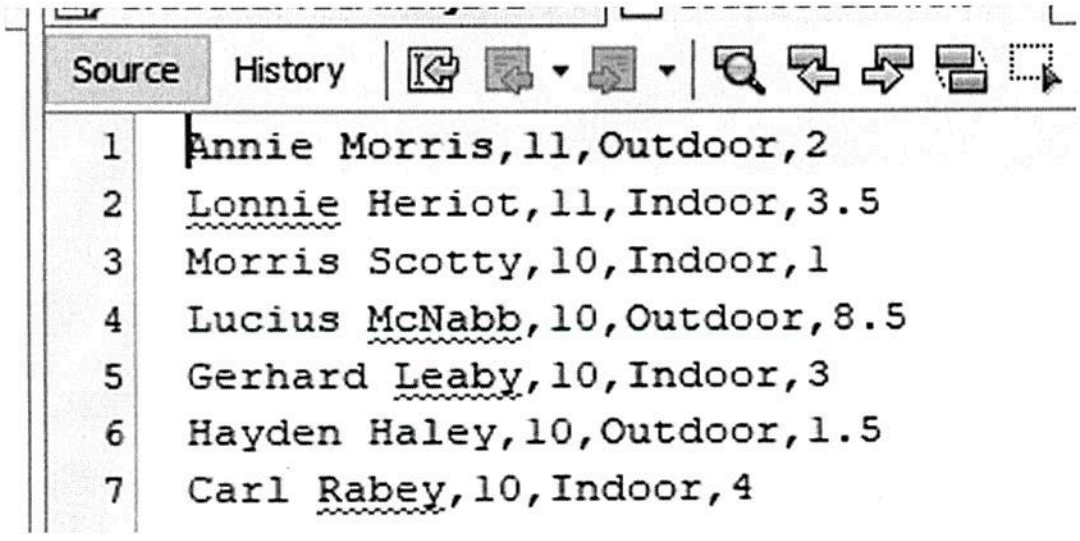
**Where to place your text file.**

Here you can see that the text file is not in the same folder as the source code (folder "src"). It is one folder above that in the Project folder.




Here is the file called **"SocialOutreach.txt"**. The fields are as follows . . .

Name, grade, Preferring outreach option i.e. outdoor, indoor or both, hours of outreach accumulated already.



To prevent the application from crashing when it does not find the text file (perhaps a typing error) we have to place some **input – output error handling code.** We have two options – "catch the error" or "throw the error". To catch the error we use "try . . . catch", which we will not use here. To throw the error we need to add some additional code and additional imports. See below . . .

```
urce  History
    // Grade 11 Practical model

    package gradellpracfinal;

    import javax.swing.*;
    import java.util.*;
    import java.io.*;

    public class Grade11PracFinal {

        public static void main(String[] args) throws IOException

        {
            String fileName;
            fileName = JOptionPane.showInputDialog("Enter name of the text file");

            Scanner scFile = new Scanner(new File(fileName));
```

Now it is time to **read in the fields** into temporary variables. The variable are temporary because they constantly get overwritten by the next record in the text file. The **size counter** tells us how many records the text file has. We will use this counter later to populate the arrays we want to use.

```
Source  History
13      {
14          String fileName;
15          double hours = 0.0;
16          int size = 0;
17          fileName = JOptionPane.showInputDialog("Enter name of the text file");
18
19          Scanner scFile = new Scanner(new File(fileName));
20
21          while (scFile.hasNext())
22          {
23              String line = scFile.nextLine();
24              Scanner scLine = new Scanner(line). useDelimiter(",");
25              String name = scLine.next();
26              String grade = scLine.next();
27              String preference = scLine.next();
28              hours = scLine.nextDouble();
29
30              size++;
31          }
32          System.out.println("There are " + size + " records in the file" );
33      }
```

```
Output - Grade11PracFinal (run)
run:
There are 30 records in the file
BUILD SUCCESSFUL (total time: 12 seconds)
```

Time to declare the **arrays** we need. We decide to have a String array for the names and a double array for the hours. By storing values in the arrays we can reuse them later.

We also need to declare a double variable so that we can total up the **hours worked**.

```
{
    String fileName;
    double totalHours = 0.0; // We want to total the hours up.
    int size = 0;
    String[] nameArr = new String[100]; // an array for names
    double[] hoursArr = new double[100]; // an array for hours
```

**Storing the hours in an array**. Printing out the array of hours. Calculating the total number of hours done in the text file.

The hour value is read into the hour array – stored in the next available array index – the hour value is added to the total hours – the while loop exits – the total hours are printed to screen – the array of hours is printed to screen one per line using a for loop. (not great, but ok for now)

```
        double hours = scLine.nextDouble();
        hoursArr[size] = hours;

        totalHours = totalHours + hours;
        size++;
    }
    System.out.println("There are " + size + " records in the file" );
    System.out.println("The total hours are " + totalHours);
    for (int i= 0; i < size; i++)
    {
        System.out.println(hoursArr[i]);
```

Now we read in the **name values** into the name array. Now we can print out the name with their hours.

```
27              Scanner scLine = new Scanner(line). useDelimiter(",");
28              String name = scLine.next();
29              nameArr[size] = name;
30              String grade = scLine.next();
31              String preference = scLine.next();
32
33              double hours = scLine.nextDouble();
34              hoursArr[size] = hours;
35
36              totalHours = totalHours + hours;
37              size++;
38          }
39          System.out.println("There are " + size + " records in the file" );
40          System.out.println("The total hours are " + totalHours);
41          for (int i= 0; i < size; i++)
42          {
43              System.out.print(nameArr[i] + "\t");
44              System.out.println(hoursArr[i]);
45
46          }
47      }
```

```
Output - Grade11PracFinal (run)
    Corine Laughn   7.5
    Amber McCalum   5.0
    Lucile Lowther  6.0
    Melba Morvel    1.0
    Jessica Lette   0.0
    Evie Galway     4.5
    Twila Leaby     7.0
```

Time to **count the preferences** of the students i.e. how many prefer "Outdoor", how many prefer "Indoor" and how many are willing to do both. We declare and initialize three integer variables (not shown). We use a simple "if construct" to count each type of preference and then print it to console.

```
            String grade = scLine.next();
            String preference = scLine.next();
            if (preference.equals("Outdoor"))
                    outdoorPreference = outdoorPreference + 1;
            if (preference.equals("Indoor"))
                    indoorPreference = outdoorPreference + 1;
            if (preference.equals("Both"))
                    bothPreference = outdoorPreference + 1;

            double hours = scLine.nextDouble();
            hoursArr[size] = hours;

            totalHours = totalHours + hours;
            size++;
        }
        System.out.println("There are " + size + " records in the file" );
        System.out.println("The total hours are " + totalHours);
        System.out.println("============================");
        System.out.println("Outdoor preference " + outdoorPreference);
        System.out.println("Indoor preference " + indoorPreference);
        System.out.println("No preference " + bothPreference);
        System.out.println("============================");
        for (int i= 0; i < size; i++)
```

put - Grade11PracFinal (run) ×

```
============================
Outdoor preference 14
Indoor preference 8
No preference 15
============================
```

On the next page is the **full listing of our one class program** – but we need a two class program with the UI handling only the input and output and the template class handling all the processing.

Therefore our template class will have

1. A method to read in the text file.
2. A method to total the number of hours in the hours-array.
3. A method to count the preferences.

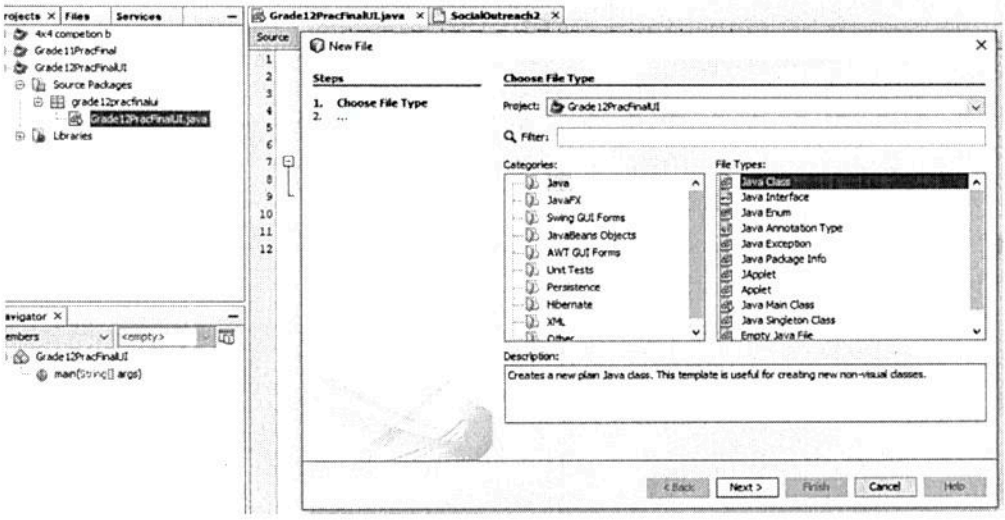
```
1 // Grade 11 Practical model
2
3 package gradellpracfinal;
4
5 import javax.swing.*;
6 import java.util.*;
7 import java.io.*;
8
9 public class Gradel1PracFinal {
10
11     public static void main(String[] args) throws IOException
12
13     {
14         String fileName;
15         double totalHours = 0.0; // We want to total the hours up.
16         int size = 0;
17         int outdoorPreference = 0, indoorPreference = 0;
18         int bothPreference = 0;
19         String[] nameArr = new String[100]; // an array for names
20         double[] hoursArr = new double[100]; // an array for hours
21
22         fileName = JOptionPane.showInputDialog("Enter name of the text file");
23
24         Scanner scFile = new Scanner(new File(fileName));
25
26         while (scFile.hasNext())
27         {
28             String line = scFile.nextLine();
29             Scanner scLine = new Scanner(line). useDelimiter(",");
30             String name = scLine.next();
31             nameArr[size] = name;
32             String grade = scLine.next();
33             String preference = scLine.next();
34             if (preference.equals("Outdoor"))
35                     outdoorPreference = outdoorPreference + 1;
36             if (preference.equals("Indoor"))
37                     indoorPreference = outdoorPreference + 1;
38             if (preference.equals("Both"))
39                     bothPreference = outdoorPreference + 1;
40
41             double hours = scLine.nextDouble();
42             hoursArr[size] = hours;
43
44             totalHours = totalHours + hours;
45             size++;
46         }
47         System.out.println("There are " + size + " records in the file" );
48         System.out.println("The total hours are " + totalHours);
49         System.out.println("============================");
50         System.out.println("Outdoor preference " + outdoorPreference);
51         System.out.println("Indoor preference " + indoorPreference);
52         System.out.println("No preference " + bothPreference);
53         System.out.println("============================");
54         for (int i= 0; i < size; i++)
55         {
56             System.out.print(nameArr[i] + "\t");
57             System.out.println(hoursArr[i]);
58
59         }
60     }
61
62 }
```

4A

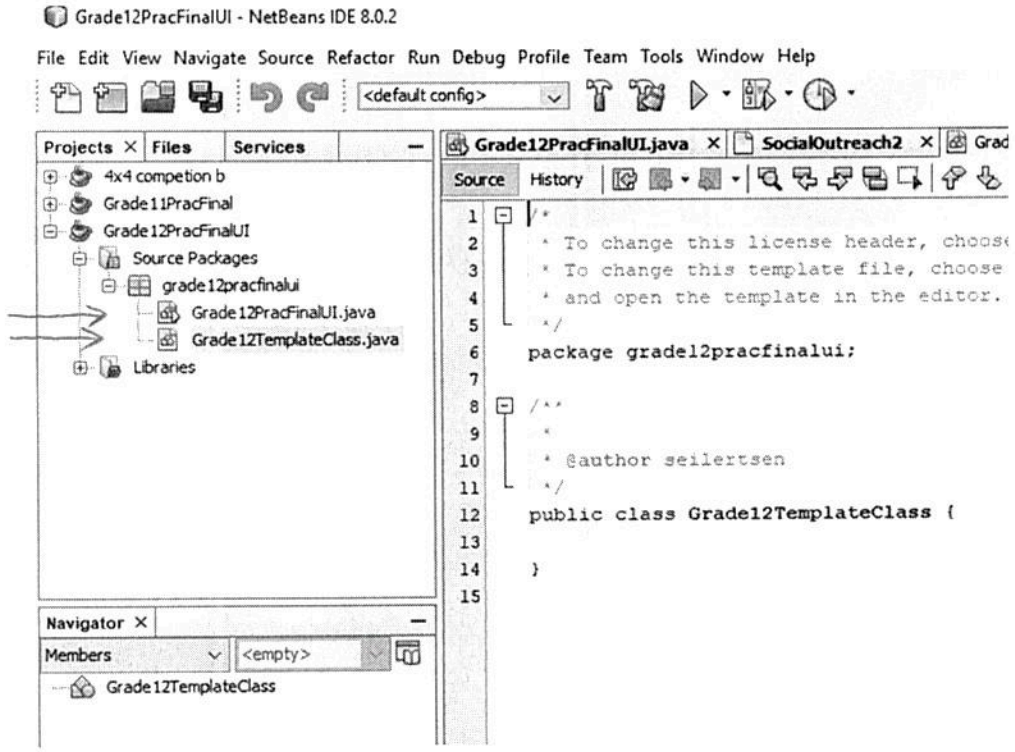
## Two Class solution of the same program

Create your first class with a main method and remember to name it "UI".

Now create a new file, a new Java Class file.




Now we have a UI class with a main method and a template class without a main method.



Once we have a template class we can **instantiate a child object** from it in the main method. Then we can start to move our code into the template class e.g. reading in the text file.

**NB: Each block of code must go into its own method. We cannot call a class – we can only call a method.**

Grade12PracFinalUI.java | Grade12TemplateClass.java

Source | History

```
1   // Grade 11 Practical two class model
2
3   package grade12pracfinalui;
4
5   public class Grade12PracFinalUI {
6
7       public static void main(String[] args) {
8
9           Grade12TemplateClass myTemplate = new Grade12TemplateClass();
10
11          System.out.println("Done");
12      }
13
14  }
15
```

**NB: Each block of code must go into its own method. We cannot call a class – we can only call a method.**

**Method One. Read in the text file**

Here below is the **main method**. It calls the "readFile" method and passes the file name. Note that the main method still needs IOException.

Source | History

```
1   // Grade 11 Practical two class model
2
3   package grade12pracfinalui;
4
5   import javax.swing.*;
6   import java.io.*;
7
8   public class Grade12PracFinalUI {
9
10      public static void main(String[] args) throws IOException {
11          String fileName;
12
13          Grade12TemplateClass myTemplate = new Grade12TemplateClass();
14          fileName = JOptionPane.showInputDialog("Enter the file name");
15          myTemplate.readFile(fileName);
16
17          System.out.println("Done");
18      }
19
```

The **readFile method** in the child object accepts the filename and reads in the text file. Note that this method also needs IOException. Names and hours are assigned into their relevant array structures.

NOTE: This can be done via the **constructor method** but we are not going to do that here.

NOTE 2: The variables that the whole class needs are **declared as private and are outside** of all the methods.

Grade12PracFinalUI.java | Grade12TemplateClass.java

Source | History

```
4
5   import java.util.*;
6   import java.io.*;
7
8   public class Grade12TemplateClass { // Declare variables for whole class
        private String[] nameArr = new String[100]; // an array for names
        private double[] hoursArr = new double[100]; // an array for hours
11      private int size = 0;
12
13      public void readFile(String f) throws IOException // Method to read in text
14      {
15          String fileName = f;
16          Scanner scFile = new Scanner(new File(fileName));
17
18          while (scFile.hasNext())
19          {
20              String line = scFile.nextLine();
21              Scanner scLine = new Scanner(line). useDelimiter(",");
22              String name = scLine.next();
23              nameArr[size] = name; // read name into name array
24              String grade = scLine.next();
25              String preference = scLine.next();
26
27              double hours = scLine.nextDouble();
28              hoursArr[size] = hours; // read hours into hours array
29
30              size++;
31          }
```

**Method Two. Determine the total number of hours**

The call in the main method. It receives a double value back – therefore assign to hours variable.

```
        // Calcalte total hours
        totalHours = myTemplate.calcHours();
        System.out.println("Total number of hours is " + totalHours);

        System.out.println("Done");
    }
```

The method in the child object. It returns a double value "totalHours"

```
34      public double calcHours()
35      {
36          double totalHours = 0.0;
37          for (int i = 0; i < size; i++)
38              totalHours = totalHours + hoursArr[i];
39
40          return totalHours;
41      } // endcalcHours
42
```

**Method Three. Determining the students preferences – outdoor, indoor or both**

We need another array to store the student preferences.

```
7
8   public class Grade12TemplateClass { // Declare variables for whole class
        private String[] nameArr = new String[100]; // an array for names
        private double[] hoursArr = new double[100]; // an array for hours
        private String[] preferenceArr = new String[100]; // an array for pref
12      private int size = 0;
13
```

The call in the main method. The method does not return any value therefore we don't need to assign a variable.

```
21              // Determine preference count
22              myTemplate.calcPreferences();
23              System.out.println("Done");
```

The method in the child object. It is void as it does not return a value. Output here in the method is not a great solution but is ok for now.

```
46      public void calcPreferences()
47      {
48          // Local variables needed here only
49          int outdoorPreference = 0, indoorPreference = 0;
50          int bothPreference = 0;
51
52          for (int i = 0; i < size; i++)
53          {
54             if (preferenceArr[i].equals("Outdoor"))
55                    outdoorPreference = outdoorPreference + 1;
56             if (preferenceArr[i].equals("Indoor"))
57                     indoorPreference = indoorPreference + 1;
58             if (preferenceArr[i].equals("Both"))
59                    bothPreference = bothPreference + 1;
60          } // end for
61
62          // Output here is not a great solution, but ok for now
63          System.out.println("Outdoor preference " + outdoorPreference);
64          System.out.println("Indoor preference " + indoorPreference);
65          System.out.println("No preference " + bothPreference);
66
67      } // end calcPreferences
```

```
Output - Grade12PracFinalUI (run)
run:
Total number of hours is 130.0
Outdoor preference 14
Indoor preference 0
No preference 0
Done
```

```
1 // Grade 11 Practical two class model
2
3 package grade12pracfinalui;
4
5 import javax.swing.*;
6 import java.io.*;
7
8 public class Grade12PracFinalUI {
9
10     public static void main(String[] args) throws IOException {
11         String fileName;
12         double totalHours = 0.0;
13
14         // Instantiate the child object from template class
15         Grade12TemplateClass myTemplate = new Grade12TemplateClass();
16
17         // Read in text file
18         fileName = JOptionPane.showInputDialog("Enter the file name");
19         myTemplate.readFile(fileName);
20
21         // Calculate total hours
22         totalHours = myTemplate.calcHours();
23         System.out.println("Total number of hours is " + totalHours);
24
25         // Determine preference count
26         myTemplate.calcPreferences();
27         System.out.println("Done");
28     }
29
30 }
```

For your PAT.

Note the use of comments!

(9)

```
1  // Template class. Two class model
2
3  package grade12pracfinalui;
4
5  import java.util.*;
6  import java.io.*;
7
8  public class Grade12TemplateClass {
9
10     // Declare variables for whole class
11     private String[] nameArr = new String[100]; // an array for names
12     private double[] hoursArr = new double[100]; // an array for hours
13     private String[] preferenceArr = new String[100]; // an array for pref
14     private int size = 0;
15
16     // Method to read in text file
17     public void readFile(String f) throws IOException
18     {
19         String fileName = f;
20         Scanner scFile = new Scanner(new File(fileName));
21
22         while (scFile.hasNext())
23         {
24             String line = scFile.nextLine();
25             Scanner scLine = new Scanner(line). useDelimiter(",");
26
27             String name = scLine.next();
28             nameArr[size] = name; // read name into name array
29
30             String grade = scLine.next();
31
32             String preference = scLine.next();
33             preferenceArr[size] = preference; // read preferences into array
34
35             double hours = scLine.nextDouble();
36             hoursArr[size] = hours; // read hours into hours array
37
38             size++;
39             scLine.close();
40         } // end while
41     } // end readFile
42
43
44
45
46
47
```

(10)

```
48      // Determines the total number of social outreach hours
49      public double calcHours()
50      {
51          double totalHours = 0.0;
52          for (int i = 0; i < size; i++)
53              totalHours = totalHours + hoursArr[i];
54
55          return totalHours;
56      } // endcalcHours
57
58      // Determines the number of student preferences for
59      // outdoor, indoor or both (no preference
60      public void calcPreferences()
61      {
62          // Local variables needed here only
63          int outdoorPreference = 0, indoorPreference = 0;
64          int bothPreference = 0;
65
66          for (int i = 0; i < size; i++)
67          {
68             if (preferenceArr[i].equals("Outdoor"))
69                      outdoorPreference = outdoorPreference + 1;
70             if (preferenceArr[i].equals("Indoor"))
71                       indoorPreference = indoorPreference + 1;
72             if (preferenceArr[i].equals("Both"))
73                      bothPreference = bothPreference + 1;
74          } // end for
75
76          // Output here is not a great solution, but ok for now
77          System.out.println("Outdoor preference " + outdoorPreference);
78          System.out.println("Indoor preference " + indoorPreference);
79          System.out.println("No preference " + bothPreference);
80
81      } // end calcPreferences
82  }
```

(11)