INFORMATION TECHNOLOGY – PAPER I

GRADE 12

JUNE/JULY 2022

Time: 3 hours **Total: 150 Marks**

Examiners: L Coetzee (Section A), H Peuckert (Section B)
Moderators: C Kader (Section A), R Viljoen (Section B), L Bothma (Section A & B)

Name: _____

PLEASE READ THE FOLLOWING INSTRUCTIONS CAREFULLY:

1. This paper consists of **19** pages. Check that your paper has the correct number of pages.

2. This question paper is to be answered using Object-Oriented Programming principles. Your program must make sensible use of methods and parameters.

3. This paper is divided into two sections. All candidates must answer both sections.

4. This paper is set in programming terms that are not specific to any particular programming language (Java/Delphi) or database (Access/MySQL).

5. Make sure that you answer the questions in the manner described because marks will be awarded for your solution according to the specifications that are given in the question.

6. Only answer what is asked in each question. For example, if the question does not ask for data validation, then no marks are awarded for it, and therefore no code needs to be written.

7. If you cannot get a section of code to work, comment it out so that it will not be executed and so that you can continue with the examination. If possible, try to explain the error to aid the marker.

8. When accessing files from within your code, DO NOT use full path names of the file, as this will create problems when the program is marked on a computer other than the one you are writing on. Merely refer to the files using their names and extensions, where necessary.

9. Your programs must be coded in such a way that they will work with any data and not just the sample data supplied or any data extracts that appear in the question paper. You are advised to look at the supplied data files carefully.

10. Make sure that routines such as searches, sorts and selections for arrays are developed from first principles, and that you do not use the built-in features of a programming language for any of these routines. You may, however, use built-in sub-routines to deep-copy arrays.

11. All data structures must be defined and declared by you, the programmer. You may not use components provided within the interface to store and later retrieve data.

12. Read the whole question paper before you choose a data structure. You may find that there could be an alternative method of representing the data which will be more efficient considering the questions that are asked in the paper.

13. You must save all your work regularly on the disk you have been given, or the disk space allocated to you for this examination.

14. If there is a technical interruption that prevents you from writing your examination, e.g. a power failure, when you resume writing your examinations, you will only be given the time that was remaining when the interruption began. No extra time will be given to catch up work that was not saved.

15. Make sure that your examination number (or name and surname if you do not have your examination number yet) appears as a comment in every program that you code as well as on every page of hardcopy that you hand in.

16. Print a code listing of all the programs/classes that you code. Printing must be done after the examination. You will be given half an hour to print after the examination is finished. Your teacher will tell you what arrangements have been made for the printing of your work.

17. You should be provided with the following two folders (in bold) and files. These files are to be used as data for the examination. Note that the database file is provided in the **MS Access format**.

    Section A:
    UkraineAssistance.mdb
    SQL AnswerSheet.rtf

    Section B:
    Deliveries.txt
    Equipment.txt

SCENARIO:

On 24 February 2022, Russia invaded Ukraine, marking a steep escalation of the Russo-Ukrainian War, which had begun in 2014. The invasion was determined to be a violation of the laws of nations by the United Nations which further condemned "all violations of international humanitarian law" against the Geneva Conventions. Many countries imposed new sanctions, which have affected the economies of Russia and the world, and provided humanitarian and military aid to Ukraine.

https://en.wikipedia.org/wiki/2022_Russian_invasion_of_Ukraine

You are tasked to develop software that can assist in the logistics and distribution of humanitarian and military aid to Ukraine.

SECTION A STRUCTURED QUERY LANGUAGE

SCENARIO (FOR QUESTION 1)

The United Nations is **an international organization founded in 1945 after the Second World War by 51 countries** committed to maintaining international peace and security, developing friendly relations among nations and promoting social progress, better living standards and human rights.

The Security Council has primary responsibility for the maintenance of international peace and security. The Security Council is a body of 15 members, five of which are permanent and have veto power: the United States, United Kingdom, France, Russia and China. The newly elected five **India, Ireland, Kenya, Mexico and Norway,** will join the other non-permanents members.

tblMembers contains the details of all the countries that belong to the United Nations.

Field Name	Data Type	Description
CountryID	Text	The unique ID of each country.
CountryName	Text	The full name of the country.
DateJoined	Date	The date that the country joined the United Nations.
SeatOnCouncil	Boolean	Whether or not the country has a seat on the Security Council.

Sample of the data in the table **tblMembers**:

CountryID	CountryName	DateJoined	SeatOnCouncil
IND50	Indonesia	1950/09/28	☐
IRA45	Iran (Islamic Republic of)	1945/10/24	☐
IRE55	Ireland	1955/12/14	☑
IRQ45	Iraq	1945/12/21	☐
ISR49	Israel	1949/05/11	☐
ITA55	Italy	1955/12/14	☐
JAM62	Jamaica	1962/09/18	☐
JAP56	Japan	1956/12/18	☐

tblSupportType contains information about the different types of donations with a generalised description.

Field Name	Data Type	Description
SupportID	Text	A unique code used to identify the specific type of support.
Type	Text	Support is categorised accordingly: Humanitarian, Military or Financial.
GeneralDescription	Text	A general standardised description of the support type.

Sample of the data in the table **tblSupportType**:

SupportID	Type	GeneralDescription
EQUI01	Military	Explosives disposal equipment
EQUI02	Military	Gas masks
EQUI03	Military	High Resulotion Satelite Imagery
EQUI04	Military	Military Equipment
EQUI05	Military	Night vision goggles
EQUI06	Military	Protective equipment for military forces

tblDonations contains information regarding donations made to the Ukraine by countries supporting their cause.

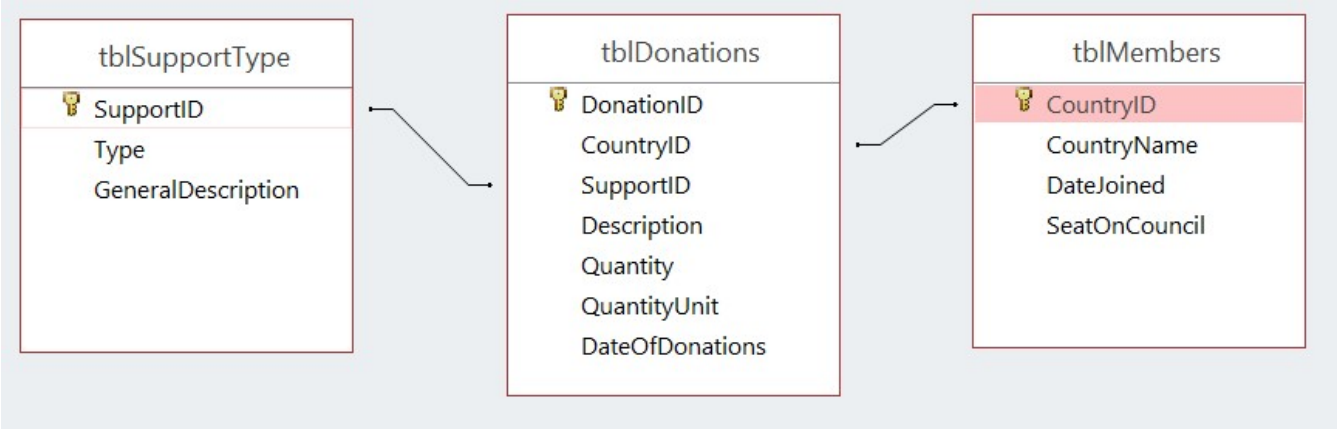
Field Name	Data Type	Description
DonationID	Integer	A unique auto-numbered identification number for each donation.
CountryID	Text	The ID of the country that made the donation. This field is a foreign key to **tblMembers**.
SupportID	Text	The ID of the type of support. This is a foreign key to **tblSupportType**.

Description	Text	A description of the donation made. This description changes from donation to donation as not all countries donate exactly the same equipment.
Quantity	Integer	The amount that has been donated.
QuantityUnit	Text	Different donations have different units of measurement. This field contains the unit of measurement.
DateOfDonations	Date	The date that the donation was made.

Sample of the data in the table **tblDonations:**

DonationID	CountryID	SupportID	Description	Quantity	QuantityUnit	DateOfDonations
1	USA45	EQUI04	Military Equipment	90	tons	2022/01/22
2	CZE93	WEAP01	152mm artillery shells	1600000	dollars	2022/01/26
3	HUN55	VEH02	Fuel	26000	gallons	2022/02/13
4	FIN55	FINA003	Non-military financial aid	86000000	dollars	2022/02/13
5	NET45	EQUI04	Combat helmets	3000	helmet	2022/02/18
6	NET45	EQUI04	Flack jackets	2000	jacket	2022/02/18

Examine the **relationships** between the 3 tables:



QUESTION 1:

Write SQL queries for each of the following. Insert your answers in the Word document named "**SQL Answersheet**".

QUESTIONS:

1.1 The United Nations was officially established on 24 October 1945. Write a query to list the first members that joined the United Nations on this date (using the **tblMembers** table). Your results should be sorted alphabetically by country name from Z to A. (4)

A sample of the correct output is shown below:

CountryID	CountryName	DateJoined	SeatOnCouncil
USA45	United States of America	1945/10/24	☑
GBR45	United Kingdom of Great Britain and Northern Ireland	1945/10/24	☑
UKR45	Ukraine	1945/10/24	☐
TUR45	Turkey	1945/10/24	☐
SYR45	Syrian Arab Republic	1945/10/24	☐
SAU45	Saudi Arabia	1945/10/24	☐
RUS45	Russian Federation	1945/10/24	☑
POL45	Poland	1945/10/24	☐

1.2 Write a query to display the names of all countries that have a seat on The Security Council. (3)

The correct output is shown on the next page.

CountryName
Russian Federation
Kenya
United Arab Emirates
Ireland
Brazil
Gabon
France
Norway
United States of America
Ghana
China
Albania
United Kingdom of Great Britain and Northern Ireland
Mexico

1.3 India (**CountryID** IND45) is a newly elected member of the Security Council but does not appear on the result set of Question 1.2. Write a query that will change India's status to reflect that they are on the Security Council. (4)

1.4 Write a query to list the names of those countries who are offering support via the Red Cross Foundation. (**SupportID** HUM09). The correct output is shown below: (4)

CountryName
Canada
Chile
China
Georgia

1.5 Different countries donate different makes/models of weapons. A **howitzer** is a long-ranged weapon, usually organised in a group called a battery. In order to plan a defence effectively the Ukraine military need to know how many howitzers they have. Write a query that will determine how many howitzers have been donated. Name this field (8)

No of Howitzers. The correct output is shown below:

No of Howitzers
2712

1.6 The president of the Ukraine, Mr Zelenskyy, would like a summary of all the donations made by country. Write a query to display a list of donations with their **general descriptions** along with the full name of the country that made the donation. (8)

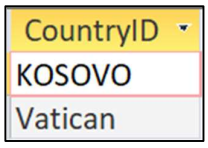
CountryName	GeneralDescription
Albania	Military Equipment
Argentina	Clothing
Argentina	Food
Argentina	Medicine
Australia	Ammunition
Australia	Amount of money donated
Australia	Assault Rifles/Guns
Australia	Food

page.

1.7 The United Nations would like to see a list of how many of each category of **humanitarian aid** donations have been made. Humanitarian donations start with **SupportID** HUM. Write a query to show how many humanitarian donations of each type has been made. Name this field **How Much**. Display the general description of the Humanitarian aid type. The correct output is shown below: (7)

GeneralDescription	How Much
Clothing	4
Field hospital	3
Firefighter and Rescue Vehicles	5
Food	7
Humanitarian Aid	16
Medical Staff	6
Medicine	13
Protective equipment for civilian	6
Red Cross	5
Refugee assistance	6
Water purifiers	1

1.8 Donations were made from countries that do not belong to the UN. Write a query to display the name of the countries that donated but do not belong to the UN. The name of each country must only appear once. The correct output is shown below: (5)



1.9 Germany has another 15 vehicles to donate. This donation is exactly the same as donation ID 220. Add another donation for Germany using the information from donation ID 220. Use today's date as the date the donation was made. (4)

1.10 Lithuania have can no longer afford to donate 10 million euro to the Ukraine Military Fund. Write a query to remove this donation from the database. (3)

50 marks

SECTION B OBJECT ORIENTED PROGRAMMING

SCENARIO (FOR QUESTIONS 2 - 8)

Many countries donate military equipment to Ukraine which is being moved forward to the Ukrainian border from neighbouring countries like Poland. For the Ukrainians to take the equipment into their war-torn country and onto the front lines is a huge logistical challenge.

You are tasked to develop an application that will assist with the distribution of weapons and vehicles into Ukraine.

Logistical information for the end of April and the beginning of May of 2022 have been captured in two text files.

The first text file called **Deliveries.txt** stores the details of the places of delivery and the dates on which the equipment must arrive at its destination.

The complete text file is shown below:

```
463105;46.74918;31.97641;2022/04/27;2;y
483208;48.77034;30.22166;2022/04/29;1;n
493811;49.41677;38.14978;2022/05/01;1;Y
483513;48.36788;35.09337;2022/05/02;1;y
483215;48.53596;32.27599;2022/05/02;1;y
463116;46.99176;31.99634;2022/05/03;1;y
463217;46.90627;32.54975;2022/05/04;4;n
493918;49.35178;39.30147;2022/05/04;4;n
463119;46.72734;31.96833;2022/05/04;3;n
473321;47.60399;33.42734;2022/05/05;3;n
463123;46.98239;31.97896;2022/05/06;2;n
473325;47.43225;33.86572;2022/05/06;4;x
503099;50.45220;30.52500;2022/12/31;0;n
```

https://www.mapsofworld.com/lat_long/maps/Ukraine-lat-long.jpg

The coordinate for the place of delivery is given in the following format in the text file:

latitude;longitude

for example:

46.74918;31.97641

Each line of the text file consists of the following information (separated by the **";"** delimiter):

- unique ID of the delivery, e.g. `463105`
- latitude of the coordinate, e.g. `46.74918`
- longitude of the coordinate, e.g. `31.97641`
- date of delivery (in the format `yyyy/MM/dd`), e.g. `2022/04/27`
- current danger level in the area of the delivery (`1 – 4`), e.g. `2`
- delivery status indicating whether the equipment has reached its destination or not (which can only be `Y`, `y`, `N` or `n`), e.g. `y`

The danger level of an area will be represented with the following values in the program:

Danger Level	Colour	Meaning
4	RED	High Danger
3	ORANGE	Considerable Danger
2	YELLOW	Moderate Danger
1	GREEN	Minor Danger

Any other danger level must be considered as UNKOWN in the program.

The second text file called **Equipment.txt** stores the details of the equipment that needs to be delivered (for the deliveries that have been listed in the first text file).

A sample of the first 18 lines of the file are listed below:

```
463105,Body Armor,1850
463105,Helmets,1850
463105,M18A1 Claymore Anti-Personnel Mines,320
463105,M113 Personnel Carriers,90
463105,Kel-Tec SUB2000 Carbine Rifles,1600
463105,9-mm Rounds,200000
483208,RQ-20B PUMA Unmanned Aerial System,1
493811,Armoured Humvees,10
493811,M119A3 Howitzers,10
493811,155-mm Rounds,50000
493811,Phoenix Ghost Tactical Drones,5
483513,AN-158 Aircrafts,2
483513,C-4 Explosives,330
483513,R7 Mako Subcompact Pistols,750
483513,9-mm Rounds,450000
483513,Kimber Tactical SOC II Rifles,560
483513,6.5-mm Creedmoor Rounds,8900
483513,Mark 4 LR/T 3.5-10x40 Rifle Scopes,70
```

Each line of the text file consists of the following information (separated by the **","** delimiter):

- the ID of the delivery the equipment belongs to, e.g. `463105`
- the name of the equipment that needs to be delivered, e.g. `Body Armor`
- the number of units to be delivered, e.g. `1850`

Example:
The first 6 lines of the text file indicate the equipment that must be delivered for Delivery ID `463105` (whose details are stored in the first text file): 1850 Body Armor, 1850 Helmets, 320 Landmines, 90 Personnel Carriers, 1600 Rifles and 200000 rounds of ammunition.

QUESTION 2

Use the class diagram below to create a new class called **Equipment**. This class will be used to store the details of a piece of military equipment that needs to be delivered. The diagram below indicates the properties and methods that are required. Take careful note of the method and parameter names in the UML diagram. **No additional property or method should be created in this class**.

Equipment

- deliveryID : integer - name : string - quantity : integer
+ Constructor (id : integer, n : string, q : integer) + getID() : integer + getQuantity() : integer + toString() : string

2.1 Create a new class named **Equipment**. (1)

2.2 Create three properties to store the ID of the delivery, name of the weapon or vehicle and the number of units that need to be delivered (as indicated in the above class diagram). (2)

2.3 Write code to create a constructor method that will accept two integers and a string as parameters and assign the values to the fields of the class. (2)

2.4 Write code to create accessor methods for the **deliveryID** and **quantity** fields of the class. (2)

2.5 Write code to create a **toString** method that will return a string representing the information of a piece of equipment in the following format:

```
<space> * <space> quantity <space> x <space> name
```

For example:

```
* 1850 x Body Armor
```

(2)

[9]

QUESTION 3

Use the class diagram below to create a new class called **Delivery**. This class will be used to create an object to store the details of one delivery. The diagram below indicates the properties and methods that are required. Take careful note of the method and parameter names in the UML diagram. **No additional property or method should be created in this class**.

Delivery
- deliveryID : integer - latitude : real - longitude : real - deliveryDate : Date - equipmentArray : Equipment [ ] - dangerLevel : string - isDelivered : boolean + LEVEL_RED : string = "High Danger" + LEVEL_ORANGE : string = "Considerable Danger" + LEVEL_YELLOW : string = "Moderate Danger" + LEVEL_GREEN : string = "Minor Danger" + LEVEL_UNKNOWN : string = "Unknown Danger Level"
+ Constructor (id : integer, lat : real, lng : real, dd : Date, dl : integer, d : char) + getID() : integer + getLatitude() : real + getLongitude() : real + getDangerLevel() : string + getIsDelivered() : boolean

```
+ getDeliveryDate() : Date

+ setEquipmentArray(equipArr : Equipment [])

+ setDangerLevel(dl : string)

+ toString() : string
```

3.1 Create a new class named **Delivery**. (1)

3.2 Write code to create the properties of the class as indicated in the above class diagram. Please note that the **deliveryDate** field must be declared as a **Date** object. (4)

3.3 Create the five constants LEVEL_RED, LEVEL_ORANGE, LEVEL_YELLOW, LEVEL_GREEN and LEVEL_UNKNOWN as shown in the class diagram. These constants must be able to be used from other classes. (3)

3.4 Write code to create a constructor method that accepts an integer **id** as a parameter representing the **deliveryID** field, a real number **lat** as a parameter representing the **latitude** field, a real number **lng** as a parameter representing the **longitude** field, a Date object **dd** as a parameter representing the **deliveryDate** field, an integer **dl** representing the **dangerLevel** field and a character **d** as a parameter representing the **isDelivered** field. Use these parameters to assign values to the fields.

Please note that you must assign the most appropriate constant to the **dangerLevel** field depending on the danger level that the method has received as a parameter. If an invalid danger level has been received, the constant of LEVEL_UNKOWN must be assigned to the **dangerLevel** field. A mark will be awarded here for code efficiency.

Also note that a Boolean value of **true** or **false** must be assigned to the **isDelivered** field depending on the character that the method has received as a parameter. If an invalid value has been received, the Boolean value of **false** must be assigned to the **isDelivered** field.

Please note that the **equipmentArray** array of objects **must not be populated** here. This will be done in Question 7.1. (7)

3.5 Create accessor methods for the **deliveryID**, **latitude**, **longitude**, **dangerLevel**, **isDelivered** and **deliveryDate** fields. Note that the **getDeliveryDate** method

must return a **Date** object. (3)

3.6 Create a setter method for the **equipmentArray** field as indicated in the class diagram. (2)

3.7 Create a setter method for the **dangerLevel** field as indicated in the class diagram. (1)

3.8 Write code to create a **toString** method that will return a string representing the information of a delivery in the following format:

```
(deliveryID) date at latitude,longitude in a dangerLevel Zone - isDelivered
```

For example:

```
(463105) 27 Apr 2022 at 46.74918,31.97641 in a Moderate Danger Zone - true
```

Please note that the delivery date must be formatted to `dd MMM yyyy`. Also, it is **not necessary** to return the data of the **equipmentArray** array at this stage. (5)

[26]

QUESTION 4

4.1 Create a new class named **DeliveriesManager**. (1)

4.2 Create two private fields as follows:

- An array of **Delivery** objects called **delivery** with enough space to store 100 **Delivery** objects.
- A counter variable called **size** to keep track of the number of **Delivery** objects. (3)

4.3 Create a constructor method that will read all the contents of the **Deliveries.txt** text file and populate the **delivery** array.

Do it as follow:

- Open the text file for reading and display a suitable error message if the file cannot be found.
- Loop through the text file and for each iteration of the loop:
  - Read a line of text from the file.
  - Split the data of the line up into separate data items.
  - Using the data items, instantiate a **Delivery** object and add it to the **delivery** array.
  - Update the counter variable called **size**. (11)

4.4 Write code to create a method called **allDeliveries.** The method must return a string that contains the information of all the deliveries by calling the **toString** method you have created in Question 3.8. Each delivery's details must be on a new line. (4)

[19]

QUESTION 5

5.1 Write code to create a text-based user interface called **LogisticsUI** that will allow simple output. (1)

5.2 Declare and instantiate a **DeliveriesManager** object. (1)

5.3 Write code to display all the deliveries' information under the heading "ALL DELIVERIES:" The equipment being delivered to these locations should not be displayed at this stage.

A sample of the output is shown below: (1)

```
ALL DELIVERIES:
(463105) 27 Apr 2022 at 46.74918,31.97641 in a Moderate Danger Zone - true
(483208) 29 Apr 2022 at 48.77034,30.22166 in a Minor Danger Zone - false
(493811) 01 May 2022 at 49.41677,38.14978 in a Minor Danger Zone - true
(483513) 02 May 2022 at 48.36788,35.09337 in a Minor Danger Zone - true
(483215) 02 May 2022 at 48.53596,32.27599 in a Minor Danger Zone - true
(463116) 03 May 2022 at 46.99176,31.99634 in a Minor Danger Zone - true
(463217) 04 May 2022 at 46.90627,32.54975 in a High Danger Zone - false
```

[3]

QUESTION 6

6.1 Return to the **DeliveriesManager** class and write code to create a new method called **getDeliveryObject** that must accept an integer as a parameter representing the ID of a delivery that needs to be searched in the **delivery** array of objects (e.g. 463217). The method must then search for that particular delivery and return the details of the delivery as a **Delivery** object. The value of **null** must be returned when the delivery was not found.

Please note that the loop must terminate when the object is found. (5)

6.2 Return to the **LogisticsUI** class and make a call to the **getDeliveryObject** method to display all the details of the delivery with ID 463217 under the heading: "DETAILS OF DELIVERY 463217:"

The correct output is shown below:

```
DETAILS OF DELIVERY 463217:
(463217) 04 May 2022 at 46.90627,32.54975 in a High Danger Zone - false
```

(2)

[7]

QUESTION 7

7.1 Return to the **DeliveriesManager** class and write code to create a new method called **populateEquipment**. Use all the data of the **Equipment.txt** text file to populate all the **equipmentArray** arrays of each object in the **delivery** array so that the equipment that needs to be delivered to the different locations is reflected. You can assume that the **Equipment.txt** file will never have more than 100 lines of text. (10)

7.2 Return to the **toString** method of the **Delivery** class (coded in Question 3.8) and add code at the appropriate places so that the equipment for each delivery will be added below it. The format of the desired output can be seen in the sample output of Question 7.4. (4)

7.3 Return to the **LogisticsUI** class and make a method call to the **populateEquipment** method so that equipment for all the deliveries can be populated. (1)

7.4 Write code in the **LogisticsUI** class to display all the deliveries' information (including all the equipment that is meant for each delivery) under the heading "ALL EQUIPMENT:"

See sample output on the next page.

A sample of the output is shown below:

```
ALL EQUIPMENT:

(463105) 27 Apr 2022 at 46.74918,31.97641 in a Moderate Danger Zone - true
 * 1850 x Body Armor
 * 1850 x Helmets
 * 320 x M18A1 Claymore Anti-Personnel Mines
 * 90 x M113 Personnel Carriers
 * 1600 x Kel-Tec SUB2000 Carbine Rifles
 * 200000 x 9-mm Rounds

(483208) 29 Apr 2022 at 48.77034,30.22166 in a Minor Danger Zone - false
 * 1 x RQ-20B PUMA Unmanned Aerial System

(493811) 01 May 2022 at 49.41677,38.14978 in a Minor Danger Zone - true
 * 10 x Armoured Humvees
 * 10 x M119A3 Howitzers
 * 50000 x 155-mm Rounds
 * 5 x Phoenix Ghost Tactical Drones
```

(1)

[16]

<SEE QUESTION 8 ON NEXT PAGE>

QUESTION 8

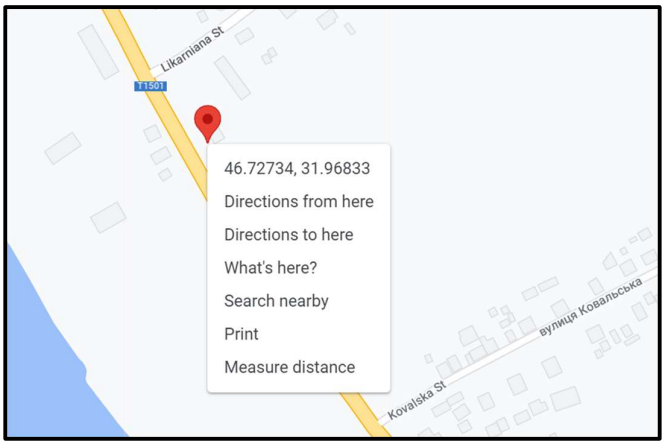
The coordinate of a location on a map is represented by two real numbers – the latitude and the longitude, e.g. for delivery ID `463119` the place of delivery has a latitude of `46.72734` and a longitude of `31.96833`.

Latitude and longitude are a system of lines used to describe the location of any place on Earth. Lines of latitude run in an east-west direction across Earth. Lines of longitude run in a north-south direction.

https://kids.britannica.com/kids/article/latitude-and-longitude/353366

https://www.mapsofworld.com/lat_long/maps/Ukraine-lat-long.jpg

So, the equipment for delivery ID `463119` needs to be delivered here:



When Russians take over a certain area, the danger levels of all the deliveries that has not yet been made in that area must be updated to RED indicating that it is a High Danger Zone.

https://ichef.bbci.co.uk/news/976/cpsprodpb/159FB/production/_124617588_ukraine_invasion_south_map-2x-nc.png

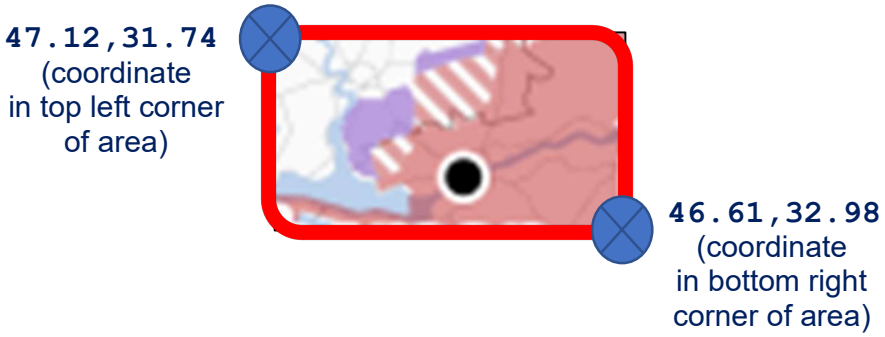
8.1 Return to the **DeliveriesManager** class and write code to create a new method called **updateDangerLevels**. This method must accept **two strings** as parameters.

The **first parameter** must represent the area that has been taken over by the Russians. You can assume that the string received will always contain two sets of coordinates in the following format:

"latitude,longitude ; latitude,longitude"

of the coordinate in top left corner of the area	of the coordinate in bottom right corner of the area

For example, the string `"47.12,31.74;46.61,32.98"` received as a parameter, indicates that the following area has been taken over by the Russians:



The **second parameter** will accept a date as a string in the format `yyyy/mm/dd`.

The danger levels of all the deliveries that have not yet been made in the newly occupied area (received as a parameter) and whose delivery dates are after the date which was received as a parameter must be updated to RED in the **delivery** array (by using the constant you have created in Question 3.3). Only do this for the deliveries that do not currently have a danger level of RED.

The method must also return the information of the deliveries that have been updated as a string in the following format before the danger level is updated:

```
deliveryID <tab> latitude,longitude <tab> danger level
```

For example:

```
463119    46.72734,31.96833    Considerable Danger
```

(18)

8.2 Return to the **LogisticsUI** class and make a method call to the **updateDangerLevels** method that will update the danger levels of the deliveries that still need to be made after `2022/04/30` in the area indicated by the string `"47.12,31.74;46.61,32.98"` to the highest possible level. You can assume that the date used as one of the arguments will always be in the format `yyyy/mm/dd`. The method call must also display the information of these deliveries (that will be updated) under the heading "ESCALATING TO HIGH RISK:".

The correct output is shown below:

```
ESCALATING TO HIGH RISK:

463119     46.72734,31.96833     Considerable Danger
463123     46.98239,31.97896     Moderate Danger
```

Please note that this list must be displayed before the “ALL EQUIPMENT” list of Question 7.4. (2)

[20]

100 marks

Total: 150 marks