Information Technology

Examiner: C. Firman

Moderator: IT Cluster **100**

Name: _____ **Time: 120 min**

READ THE FOLLOWING INSTRUCTIONS CAREFULLY

1. This paper consists of *** pages. Please check that your question paper is complete.
2. This question paper is to be answered using Object Oriented Programming principles. Your program must make sensible use of methods and parameters.
3. 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.
4. Only answer the question that is stated. For example, if the question does not ask for data validation, then no marks are awarded for it,
5. 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.
6. Read the whole question before you choose a data structure. You will find that there may be an alternative method of representing the data that will be more efficient, considering the questions that are asked in the paper.
7. You must save all your work regularly on the disk you have been given, or the disk space allocated to you for this examination.
8. If there is a technical interruption that prevents you from writing your examination, such as a power failure, when you writing your examinations, you will only be given the time that was remaining when the interruption began.
9. Print a code listing of all the programs/ classes that you code. Print the output from your program, if possible. Printing must be done after the examination. You will be given half an hour to print after the examination is finished.

SCENARIO: A fashion influencer wants to store details about all her jeans in an application. You are going to create an object class and application to store and display the following information about the jeans: brand, colour, date of purchase and number of pockets.

The influencer has decided that output in black and white is very bland and boring. She has asked that you print some of the output in colour.

Colour Printing on Screen in Java:

In order to print in colour using the System.out.println() method,

you can add a formatting string in front of the string to be printed:

"\033[34m" will print in BLUE

"\033[35m" will print in MAGENTA

"\033[36m" will print in CYAN

Example:

System.out.println("\033[36m" + “Hello there”); // Will print in CYAN (turquoise):

```
run:
Hello there
BUILD SUCCESSFUL (total time: 0 seconds)
```

Question 1: Jeans **(45 marks)**

Class Diagram:

Jeans
- brand : string - numPockets : integer - dateOfPurchase : string - colour : string + RED = "\033[31m" : string + MAGENTA = "\033[35m" : string + CYAN = "\033[36m" : string - totalJeans : integer
+ Constructor (b : string, n : int, d : string) + getBrand() : string + getNumPockets() : integer + getColour() : string + setBrand(b : string) + setNumPockets (n : integer) + setColour(c : char) + getTotalJeans() : integer + toString() : string

1.1 Code the Jeans class shown in the class diagram above. Add the following properties which will not be accessible from outside the class:

- brand of the jeans,
- number of pockets,
- date of purchase,
- colour of the jeans [6]

1.2 Code three constants for storing text. Name them: **BLUE**, **MAGENTA** and **CYAN.** [4]

1.3 Code the variable called **totalJeans** which will store the quantity of Jeans objects that have been instantiated. [1]

1.4 Code a parametrised **constructor** method to give values to the four instance variables, using the parameters shown in the class diagram. Set the colour variable to a default colour of your choice by setting it equal to one of the
static constants (RED, MAGENTA or CYAN). Increment the variable totalJeans by 1. [5]

1.5 Create **accessor** and **mutator** methods for the instance variables (except for the setColour() method which you will do in the next question). Use the parameters in the class diagram. [5]

1.6 Code an accessor method for totalJeans. [1]

1.7 Create a method called **setColour** which will accept a character as a parameter (B, M or C), and set the colour variable to be equal to one of the three static final variables according to the parameter: BLUE, MAGENTA or CYAN. [6]

1.8 Create a **toString** method which will return a String in the format shown below:

<JEANS BRAND: >brand<newline>

<POCKETS: >numPockets<newline>

<DATE OF PURCHASE: >dateOfPurchase<newline>

EXAMPLE:

For a brand of LEVI, with 5 pockets , and date of purchase: 2023-01-01 the returned String will print as follows:

```
JEANS BRAND: LEVI
POCKETS: 5
DATE OF PURCHASE: 2023-01-01
```

[4]

1.9 Create a method called ageOfJeans which will return the age of the jeans in years. Work out the age of the jeans using the dateOfPurchase and today's date. The method must return this result. (You will need to use the LocalDate and Period classes). [7]

1.10 Create a method called **flowers(int n)**. It must return a String which when printed will output some flower symbols n times in the colour of the Jeans object.

Include the following on each line:

a) The **colour** or the Jeans object, plus

b) the flowers:

✿ ❀ ❁ ✾ ✻ ✼

OR

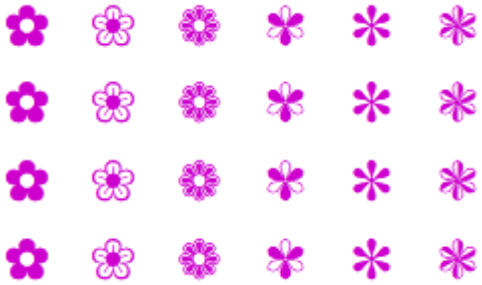
@)-,--

(Copy from here and paste into your program).

NOTE: ***The colour must be in each line to be printed.***

Example:

If n = 4 and the instance variable, colour = MAGENTA the string will print as follows when the flowers(int n) method is called from the application class:



OR if you use symbols:

@)-,--
@)-,--
@)-,--
@)-,--

[6]

Question 2 JeansAppUI **(10 marks)**

Create an application class which will:

2.1	Input the brand of the jeans and the date of purchase as a string in the correct format.	[1]
2.2	Input the number of pockets.	[1]
2.3	Instantiate a Jeans object using the values in 2.1 and 2.2.	[2]
2.4	Call the *toString()* method and display the result.	[1]
2.5	Call the *ageOfJeans()* method and display the result with a suitable message.	[1]
2.6	Input the colour as a character: R for red, M for magenta and C for cyan. Ensure the character is in upper case.	[2]
2.7	Call the *setColour(char c)* method with the value inputted in 2.5.	[1]
2.8	Call the *flowers()* method with the number of pockets as the parameter and output the result.	[1]

SAMPLE OUTPUT:

For input: brand = “Madewell”, pockets = 4, date of purchase = “2022-04-20”

```
JEANS BRAND: Madewell
POCKETS: 4
DATE OF PURCHASE: 2022-04-20
AGE OF JEANS: 1
@)-,--
@)-,--
@)-,--
@)-,--
```

Question 3 (45 marks)

This application will read from the text file: Airports.txt.

The text file is shown below:

Airports.txt

```
Beijing Capital International Airport%Beijing%China%34513827
Shanghai Hongqiao International Airport%Shanghai%China%31165641
Hartsfield-Jackson Atlanta International Airport%Atlanta%United States%42918685
Chengdu Shuangliu International Airport%Chengdu%China%40741509
Dallas/Fort Worth International Airport%Grapevine%United States%39364990
Guangzhou Baiyun International Airport%Guangzhou%China%43767558
Shenzhen Bao'an International Airport%Shenzhen%China%37916054
Chongqing Jiangbei International Airport%Chongqing%China%34937789
Denver International Airport%Denver%United States%33741129
Kunming Changshui International Airport%Kunming%China%32990805
```

Each line of the file contains 4 pieces of information separated by % signs:

Airport name % City % Country % Number of Visitors per Year

3.1 Create an application class called: AirportsApp [1]

3.2 Open the Airports.txt file for reading. If the file cannot be found, then display an appropriate message to the user. The program must not crash. [3]

3.3 Create a loop which will allow you to read all the lines of data from the text file. The loop must end when there is no longer data left to read. The object for reading from the file must be closed at the appropriate place in the code. [2]

3.4 For each line that is read from the file, store the data in four variables with meaningful names and correct types according to the data they contain. [6]

3.5 Display the data neatly as shown below (first 5 rows of data shown):

```
AIRPORT                                        CITY          COUNTRY          NUMBER OF
Beijing Capital International Airport          Beijing       China            34513827
Shanghai Hongqiao International Airport        Shanghai      China            31165641
Hartsfield-Jackson Atlanta International Airport Atlanta     United States    42918685
Chengdu Shuangliu International Airport        Chengdu       China            40741509
Dallas/Fort Worth International Airport        Grapevine     United States    39364990
```

Note: the headings must be printed at the appropriate place in the code. [5]

3.6 Using tabs to separate the printed data does not produce neat columns as some airport names, cities and countries are much longer than others. Code a static method called **spaces(String s, int n)** in the AirportsApp class. It must return a String.

This method must take the parameter String s, pad it with spaces and return a String that has a length of n. HINT: the method must add (n – s.length()) spaces to s and return this.

Call this method at the appropriate places in the main method so that the airport names, cities and countries are of the correct length when printed (padded with spaces). Ensure that the data is printed in neat columns below the headings. [8]

3.7 Add code to calculate the average number of visitors that have visited these airports in one year and display this with a meaningful message. This number must be rounded off to no decimal places and stored as an integer. [6]

3.8 Add code to find the most popular airport (with the most visitors) in the list of airports. Display this airport with a meaningful message. [6]

3.9 Add code to create a helper method called numWithCommas(int n) to convert the integer parameter to a String with thousands separators. The method must return a String. EG the number 45989349 will be displayed as 45,989,349. The following algorithm can be used:

```
temp ← convert n to a string
newStr ← empty string + character at end of temp
for loop from (temp.length – 2)  to 0   decrease by 1
        if ( loop -1) is divisible by 3
                newStr ← character at loop + “,” + newStr
        else
                newStr ←character at loop + newStr
        end if
  end for loop
  return newStr
```

In the main method, call the *numWithCommas(int n)* method when printing the average number of visitors so that it displays with thousand separators. [8]

OUTPUT:

Your output should look as shown below. (Values must not be hard-coded. They must be read from the text file / calculated).

```
AIRPORT                                          CITY            COUNTRY          NUMBER OF PASSENGERS
Beijing Capital International Airport             Beijing         China            34513827
Shanghai Hongqiao International Airport           Shanghai        China            31165641
Hartsfield-Jackson Atlanta International Airport  Atlanta        United States    42918685
Chengdu Shuangliu International Airport           Chengdu         China            40741509
Dallas/Fort Worth International Airport           Grapevine       United States    39364990
Guangzhou Baiyun International Airport            Guangzhou       China            43767558
Shenzhen Bao'an International Airport             Shenzhen        China            37916054
Chongqing Jiangbei International Airport          Chongqing       China            34937789
Denver International Airport                      Denver          United States    33741129
Kunming Changshui International Airport           Kunming         China            32990805
The average number of visitors: 37,205,798
The airport with the most visitors is: Guangzhou Baiyun International Airport
```