



eduresourceza.com

Information Technology Practical

SECTION A Object Oriented Programming 70 Marks

INTRODUCTION

Jason helps his dad to sell paintings at the flea market. His dad would like to keep track of the paintings that he has. He has created a text file called **paintings.txt**. The file contains values for the Artist, Medium, Style and Price separated by a # as shown below:

```
Tracy Dawson#watercolour#Realism#R2800.00#  
Sky Jones#oil#Cubism#R3000.00#  
Ian Finlay#graphite#Baroque#R2300.00#  
Mark Ross#acrylic#Abstract#R1550.00#  
Sky Jones#watercolour#Realism#R3200.00#  
Fred Flintstone#graphite#Cubism#R5.00#  
Ian Findlay#watercolour#Baroque#R10000.00#  
Tracy Dawson#acrylic#Abstract#R2000.00#  
Sky Jones#oil#Cubism#R2000.00#  
Mark Ross#acrylic#Abstract#R37550.00#
```

His stall can have a maximum of 50 paintings.

Question 1

- 1.1. Create a class called **Painting** with 4 instance variables to store the artist, medium, style and price of each painting. Use a suitable data type for each of these variables and ensure that these variables cannot be accessed outside of this class. (4)
- 1.2. Code a parameterised constructor method to assign values to these variables. (4)
- 1.3. Code 4 accessor methods (get), one for each variable and a mutator method (set) to modify the price variable's value. Give these methods appropriate names. (5)
- 1.4. Create a **toString()** method that will display the fields of this class neatly in columns on the screen. Use a separate method or tabs to align your columns. (4)

[17 Marks]

Question 2

- 2.1. Create a class called **ManyPaintings**. (2)
- 2.2. Declare an array **paintingArr** as an instance variable that will store objects of the **Painting** class. (3)
- 2.3. Declare an instance variable called **size** for the actual number of paintings in the array. (2)

- 2.4. Code a constructor method that will instantiate the array using the data from the **paintings.txt** file. (8)
[15 Marks]

Question 3

- 3.1 Create a class **PaintingApp** that will display a menu with the option **OOP**. Under this menu option list the following menu items: (2)

OOP

Display Paintings
Search
Display above ave
Generate code

- 3.2 For each menu option in the **PaintingApp** class, call a method in the **ManyPaintings** class, which must do the following :

- **OOP | Display Paintings:**
Display each painting's details, in columns, under appropriate headings. (6)

- **OOP | Search :**
The user must be prompted to input the name of an artist and the medium. Use these as search fields to find all paintings by that artist and the medium requested. Display only the name of the artist and the price(s) of the painting(s) under appropriate headings. If no such artist is found, display a suitable message.

Sample input :

```
Enter the artist      : Sky Jones
Enter the medium     : oil
```

Sample output :

```
Sky Jones           R5000.00 (10)
```

- **OOP | Display above ave:**
Code a method **priceGreater** to list all the paintings whose price is greater than the average price of all the paintings. This method must call a typed method called **calcAve** which will calculate the average price of the paintings and return this amount. (8)

- **OOP | Generate code:**
Code a method to generate a code for each painting as follows : the first 3 letters of the artist's surname (in lowercase) must be joined to the first character of the artist's first name (in lowercase) followed by the style of the painting with all vowels removed. Since many artists work in the same style, add two random digits to the end of the code.

eg. The code for Ian Finlay#graphite#Baroque#R2300.00
must become finibrq63 assuming 6 and 3 were generated.

(5 Marks will be awarded for placing code in the correct classes and following good
Object Oriented Principles). (12)
[38 Marks]

SECTION B SQL 50 Marks

INTRODUCTION

A parents fundraising committee have been selling food and drinks at schools events. The sales have been recorded for each week with the parents name, what they were selling (beer, food or cold drink) and how many items they sold. The information is stored in a database called **Sales** in a table named **tblDrinks**. A copy of the table is shown below. The field names are given as column headings.

tblDrinks

NAME	WEEK	ITEM	AMOUNT
Joe	3	Beer	18
Beth	4	Food	17
Janet	5	Beer	14
Joe	3	Food	12
Joe	4	Beer	19
Janet	5	Cold drink	12
Joe	3	Food	19
Beth	4	Beer	15
Janet	5	Beer	19
Joe	3	Cold drink	20
Joe	4	Beer	16
Beth	5	Food	12
Beth	3	Beer	16
Joe	4	Food	17
Joe	5	Beer	14
Janet	3	Cold drink	19
Joe	4	Food	17
Beth	5	Beer	20
Janet	3	Food	18
Joe	4	Beer	14
Joe	5	Food	12
Janet	3	Beer	18
Janet	4	Cold drink	17
Janet	5	Food	12

1. Add the following menu items from a new menu option called **SQL** to the **PaintingApp**.

OOP

Display Paintings

Search

Display above ave

Generate code

SQL

View all sales

Add a sale

Sales by person

Total sales

Sales per type

(2)

[2 Marks]

2. Open the **DB** class that has been provided to you. Add this file into your project.
- Code a method in the DB class called **query** to allow a query on the Sales database. The method must accept a SQL statement as a parameter and return the result of the query. (5)
 - Code a second method called **update** to edit or change a table in the database given a SQL parameter. (5)
 - Add a method to convert a Java result set object into a String. This method must be used in conjunction with the query method so that the query method returns a String to the calling class. (5)
- [15 Marks]

3. Use the methods you coded in question 2 to code solutions for the following menu options:
- View all sales**
Add code to show all the sales in the table ordered by name, item then week. (7)
 - Add a sale**
Add code to add a new sale. Input all the fields required from the user and add the record to the table. Before you insert the data ensure that the record you are about to enter is not a complete duplicate of an existing record. If the exact record is found in the database, display an error message and do not insert the record. (10)
 - Sales by person**
Add code to find all the sales made by a particular person. Input the person's name and then display their name, items and amount sold. (6)
 - Total sales**
Add code to display the total sales of all the items. (5)
 - Number sales per type**
Add to code to display the total number of items sold made per item. For example 183 items of beer were sold. (5)
- [33 Marks]