MULTIMEDIA UNIVERSITY

STUDENT ID NO | | | | | | | | | | |

MULTIMEDIA UNIVERSITY

FINAL EXAMINATION

TRIMESTER 1, 2019/2020

DCS5088 – OBJECT ORIENTED PROGRAMMING

(For DIT students only)

16 OCTOBER 2019
2.30 p.m – 4.30 p.m
(2 Hours)

INSTRUCTIONS TO STUDENTS

1. This examination paper consists of **11** pages.
2. **SECTION A:** There are **3** structured questions.
3. **SECTION B:** There is **1** structured question.

SECTION A: Structured Questions (Total: 30 Marks)

Instruction: Please write all your answers in the Answer Booklet provided.

QUESTION 1 (10 Marks)

a. Complete the following program (starting from the comment "***//1a)----------***") that will :

- Display a series of numbers "1 4 9 16 25 36 49 64 81 100 ". Use built-in math functions where necessary.
- Determine the occurrences of odd numbers based on the series. [Tip: use variable *count* to increase the occurrences if the number is an odd number]
- Display the occurrences. [4 marks]

```
#include<iostream>
#include<cmath>
using namespace std;

int main()
{
    int i = 1, count = 0;
    float result;

    while(i<=10)
            //1a)-----------

  return 0;
}
```

SAMPLE OUTPUT SCREEN

```
1 4 9 16 25 36 49 64 81 100
There are 5 odd numbers in the series.
```

b. Write the program segments based on the descriptions given:

i) Create a structure called *Foods*. The data members are:

- *orac_score* ***(int)***
- *name****(character array, size 30)*** [1 mark]

ii) Declare an object name *f1* of struct *Foods*. Set the *orac_score* to 8100. [1 mark]

Continued...

c. Given the program below:

```
#include <iostream>
using namespace std;

class Purchase
{  int qty;
   float price;

  public:
      void setdata(int x, float y)
   {    qty = x;
        price = y;
       }

   float calculate()
   { return qty*price; }
};

int main()
 {
     Purchase A, B;
     float tot, discount ;

     A.setdata(10, 10);
     B.setdata(1, 50);

     A.setdata(10, 20);
     tot = A.calculate() + B.calculate();
     cout<< "The total is : RM" <<tot<<endl;

     // 1c. iii) ----------------

     return 0;
 }
```

i) List down all the data members of *Purchase* class. [0.5 mark]

ii) List down all the member functions that *Purchase* class has. [0.5 mark]

iii) Write the codes to complete segment labelled '***//1c. iii)-------***'. Use *if-else if* statements to determine the discount based on the total. Display the discount and the discounted total. [3 marks]

Continued...

Total	Discount
>=0 and <100	2%
>= 100 and < 200	5%
>=200	10%
None of the above	0%

SAMPLE OUTPUT SCREEN

```
The total is : RM250
You will get 10 %
After discount, is RM225
```

QUESTION 2 (10 Marks)

a. Given the program below:

```
#include<iostream>
#include<iomanip>
using namespace std;

class Bag
{    private: string brand;
              char code;
              float price;
     public:

          void setData( string b, char c)
          {  brand = b;
             code = c;
          }

          float getPrice()
          {   // 2a. i) ----------

          }

         float display()
         { float pr = getPrice();
             cout << "\nBag brand       : " << brand << endl;
             cout << "Bag code        : " << code << endl;
             cout << "Bag price       : RM " << price << endl<<endl;
             return pr;
         }

         // 2a. ii) -----------

};
```

Continued...

```
class Order
{  Bag bb;
   int qty;
   public:

   // 2a. iii) ----------

};
int main()
{  Order P(5);    }
```

SAMPLE OUTPUT SCREEN

```
New order !
Enter bag brand     :MelC
Enter bag code      :N

Bag brand        : MelC
Bag code         : N
Bag price        : RM 2000

===Grand total is RM 10000.00 for 5 bags
--------end of program-------
```

i) Write the codes to complete segment labelled '*//2a. i)-------*'. Complete the member function *getPrice()* of *Bag* class. The function has to determine the *price* based on the *code*. Use switch-case block to assign the price based on the given table. After the switch-case block, return *price*. [2 marks]

Code	Price
X or B	1000
C or N	2000
V	5000
None of the above	set price to 0 and display "Invalid Code"

ii) Write the codes to complete segment labelled '*//2a. ii)-------*'. Write a destructor for Bag class that outputs "*----end of program--*". [1 mark]

iii) Write the codes to complete segment labelled '*//2a. iii)-------*'. Write a parameterized constructor of *Order* class with one argument that initializes the *qty* and does the following:

Continued...

- Get user input for *brand* and *code*. Then, using object *bb* call the *setData(...)* function passing the brand and code.
- Get price of object *bb* by calling *display()*
- Display the grand total.
  [Note: refer to sample output screen above]

[2.5 marks]

b. Given the program below :

```
#include<iostream>
#include<iomanip>
using namespace std;

class Temperature
{  protected:
     double degrees;
     char scale; //'F' for Fahrenheit or 'C' for Celcius
public:
   void set(double d, char s)
   { degrees = d;
     scale = s;
   }
};

class OvenTemperature : public Temperature
{  string category;
   public:
      void displayCategory()
      {
         //2b. ii)-------------

         cout<<degrees<<scale<<" "<<category<<endl;
      }
};

int main()
{   double d[6] = {330, 222, 155, 110, 323, 170};
    char scale[6]={ 'F', 'F', 'C', 'C', 'F','C' };

     //2b. iii) -------------

}
```

i) Identify the base class and derived class. [0.5 marks]

ii) Complete segment labelled '***//2b. ii)*** *-------*' which completes member function named *displayCategory()* of *OvenTemperature* class. Use **nested *if*** statements to determine the category based on the given table. [2 marks]

Continued...

Scale	Degrees	Category
F	below 300	"Very Slow"
	>= 300 and less than 325	"Slow"
	Other values	"Others"
C	Below 150	"Very Slow"
	>= 150 and less than 165	"Slow"
	Other values	"Others"
-	-	"Invalid"

iii) Complete segment labelled '***//2b. iii)-------***' by writing the codes to do the following:

- Declare a *dynamic* array of *OvenTemperature* objects (size 6).
- Using any looping structure, loop through all the object elements to call *set(...)* and *displayCategory()*. Pass the array element of *d* and array element *scale* from the main() as *set(...)*' s parameter list.
- Deallocate the memory of the dynamic array.

[2 marks]

[**Note:** Refer to sample output given below]

SAMPLE OUTPUT SCREEN
`330F others`
`222F very slow`
`155C slow`
`110C very slow`
`323F slow`
`170C others`

Continued...

QUESTION 3 (10 Marks)

a. Given the ***main()***, create a ***Coordinate*** class to contain suitable data members and member functions including member functions that overloads the addition and subtraction operators. Include suitable constructors (default and parameterized constructor definitions) in the class as well. [4.5 marks]

```
int main()
{
    Coordinate p1(4.0, 10.0);
    Coordinate p2(6.0, 3.0);
    Coordinate p3, p4;

    p4 = p1 + p2 ;
    p3 = p1 - p2 ;

   return 0;
}
```

b. Observe the program below:

```
#include<iostream>
using namespace std;
class Employee
{    protected:
        float salary;

     public:
        virtual void display ( float s )
        {  salary = s;
           cout<< "= Your salary is :: RM " << salary<< endl;
        }

        virtual ~ Employee ()
        { cout<< "Employee class" << endl << endl; }
};

class Manager: public Employee
{     void display( float s )
      {  salary = s;
         cout<< "This is a Manager's salary :" << salary << endl;
      }

      ~Manager()
      { static int num=1;
        cout << "Manager " << num << " salary calculation done" <<
        endl ;
```

Continued...

```
        num++;
      }
};
int main()
{   Employee j;
    Employee *p = new Manager;
    p->display(2590);
    delete p;

    p = new Manager;
    p->display(3670);
    delete p;

    p = &j;
    p->display(1000);

   return 0;
}
```

i) Trace the output produced by the program above. [2.5 marks]

ii) Rewrite *display()* of *Employee* class as a pure virtual function. [1 mark]

iii) Identify the errors if main() is replaced with new code as below. Copy the erroneous line(s) and correct it. [2 marks]

```
//updated main()
int main()
{
    Manager *p = new Employee;
    p->display(5000);
    delete p;

    p = new Manager(12000);
    p->display(11000);
    delete p;
    return 0;
 }
```

Continued...

SECTION B (Total: 20 Marks)

Instruction: Please write all your answers in the Answer Booklet provided.

Write a **complete program** with three classes as described below.

[**Note:** Refer to sample output given below. The ***bold items*** are inputs entered by user.]

➢ Create class called ***Author*:**

- Protected data members:
  - *name* : data type string
  - *email* : data type string
- Public member functions:
  - ***setData(....)*** : -Sets the value for *name* and *email*.

➢ Create class called ***Book*** [**derived publicly** from class ***Author***]:

- Private data members
  - *publisher* : data type string
  - *title* : data type string
  - *price* : data type double
- Public member functions:
  - ***getInput( )*** : - Prompt user to enter value for book *title*
    - Prompt user to enter value for *publisher* name
    - Prompt user to enter value for *price*
  - ***display( )*** : Display author's name, author's email, book's title, book's publisher and book's price.
  - Declare *Reorder* class as a friend of *Book* class

➢ Create class called ***Reorder*:**

- Public data members
  - *total* : data type double
- Public member functions:
  - *constructor* : - initialize total with 0

Continued...

- Define a function that overloads the "**+=**" operator. The overload function takes in an argument called *b* of class *Book*. Sum up the *price* of the *b* object in *total*. Display the *title* of *b* object.

In ***main()***:

1. Declare the necessary variables
2. Declare *ob*, an object of Reorder class.
3. Create pointer *b* of class *Book*
4. Use the pointer *b* to create a dynamic object of *Book* class using new operator.
   a. Prompt user to enter the author's name and email.
   b. Call *setData(.....)*, passing the author's name and email as the functions' arguments.
   c. Call *getInput()*
   d. Call *display()*
   e. Call the overloaded operator function (+=) using *ob*, passing the dynamic object as the parameter.
5. Deallocate memory for the dynamic object.
6. Repeat step 4 and 5 for another dynamic object.
7. Display *total* of *ob* object.

SAMPLE OUTPUT SCREEN

```
---------------------------------
-------     Get Input      -------
Enter Author name                : Edward Schneierman
Enter Author's email             : edward@mit.edu.my
Enter book title                 : C++ for Mathematicians
Enter publisher name             : Chapman & Hall
Enter price                      : 40.50

-------------End of Input-------

---------------------------------
-------   Book Details    -------
---------------------------------
Author name      :Edward Schneierman
Author email     :edward@mit.edu.my
Publisher        :Chapman & Hall
Book title       :C++ for Mathematicians
Price(RM)        :40.50
reordering a copy of C++ for Mathematicians

---------------------------------
-------     Get Input      ------
Enter Author name                : Haris Iskandar
Enter Author's email             : haris@city.edu.my
```

Continued...

```
Enter book title                : Design of UI
Enter publisher name            : MCorp
Enter price                     : 88.80

-------------End of Input--------

-----------------------------------
-------   Book Details    -------
-----------------------------------
Author name      :Haris Iskandar
Author email     :haris@city.edu.my
Publisher        :MCorp
Book title       :Design of UI
Price(RM)        :88.80
reordering a copy of Design of UI
Reordering a copy of each book will cost a total of RM 129.30
```

End of Page.