University of Colombo, Sri Lanka

University of Colombo School of Computing

BACHELOR OF SCIENCE IN COMPUTER SCIENCE

First Year Examination — Semester II– UCSC AY19 [held in March/April/May 2023]

SCS1209 — Object Oriented Programming

(Two (2) Hours)

74

Answer ALL questions

Number of Pages = 14 Number of Questions = 4

To be completed by the candidate								
Index Number								

Important Instructions to candidates:

- Students should answer in the medium of English language only using the space provided in this question paper.
- Note that questions appear on both sides of the paper. If a page is not printed, please inform the supervisor immediately.
- Write your index number CLEARLY on each and every page of this question paper.
- The duration of the paper is **Two (2)** hours.
- This paper has 4 questions in **14** pages (including the Cover Page).
- Answer **all** the questions.
- Each question carries exactly **25 marks**.
- Calculators and any electronic device capable of storing and retrieving text including electronic dictionaries, smart watches and mobile phones are not allowed.
- Do not tear off any part of this question paper. Under no circumstances may this paper, used or unused, be removed from the Examination Hall by a candidate.

To be completed by the examiners

1	
2	
3	
4	
Total	

Index Number | | | | | | | | |

1. (a). Briefly explain the term "Object-Oriented Programming".

[2 marks]

(b). List down three (3) core principles supported by Object Oriented Programming.

1.)

2.)

3.)

[3 marks]

(c). State whether the following statements are TRUE or FALSE. In either case, justify your answer.

[10 marks]

i. The member variables and functions defined in both a `structure` and a `class` are visible to all functions within its scope by default.

ii. The overloaded functions listed below are valid in C++.

```
float MyFunction(int x, String y);
double MyFunction(int x, String y, double z=2.3);
```

Index Number | | | | | | | | |

iii. Scope resolution operator : : cannot be overloaded because it is used to specify the scope of a function or variable.

iv. Friend function in C++ is a function that breaks the concepts of encapsulation and data hiding, enabling the non-member functions to access an object's private or protected data.

v. The assignment operator creates a new object as a copy of an existing object, while the copy constructor modifies an existing object to have the same values as a new object, creating a separate memory block for the object.

(d). Write down the console output of the below programs written in C++.
In case of syntactical errors, indicate the error statement/s in the program using a Box and mention the output as a `compilation error`. In such circumstances, briefly explain how to correct the code.
Assume that all the other required lines of codes are in place to run programs.

[10 marks]

Index Number | | | | | | | | |

i.
```
class Example {
public:
  void setValue(int value) {
    value=20;
    this->value = value; }
  void printValue(){
    cout<<"Value = " <<value<<endl;}
private:
  int value;
};
int main() {
    Example ex;
    ex.setValue(42);
    ex.printValue();
}
```

.....

ii.
```
const int count =100;
class ExStatic {
  public:
    static int count;  };

int ExStatic::count = 0;

int main() {
  ExStatic ex1, ex2;
  ExStatic::count++;
  ex1.count++;
  ex2.count++;
  cout << "::count = "<<::count<<endl;
  cout << "ExStatic::count = "<<ExStatic::count<<endl;
  cout << "ex1.count = "<<ex1.count<<endl;
  cout << "ex2.count = "<<ex2.count<<endl;
}
```

.....

.....

.....

.....

Index Number | | | | | | | | |

iii.
```
class Person {
  private:
    string name;
    int age;
  public:
    Person(const Person &other) {
      cout<<"Inside the copy constructor"<<endl;
      this->name = other.name;
      this->age = other.age;
    }
    Person(string name, int age) {
      cout<<"Inside the normal constructor"<<endl;
      this->name = name;
      this->age = age;
    }
};
int main() {
  Person per1("Mala", 30);
  Person per2 = per1;
  Person per3,per4;
}
```

.....

iv.
```
class Number{
  int x,y;
  public:
    Number():x(0),y(0){}
    void operator ++() {
      x=x+100;
      y=y+100;  }

    void display(){
      cout<<"X is "<<x;
      cout<<"\nY is "<<y<<endl; }
};

int main(){
  Number N1,N2;
  N1.display();
  ++N1;
  N2++;
  N1.display();
  N2.display();
}
```

.....

Index Number | | | | | | | | |

2. Consider the following Code segment written in C++ to add two Distances.

```
class Distance {
  private:
    int feet;
    float inches;

 public:
    Distance() : feet(0), inches(0.0) {}

    Distance(int ft, float in) : feet(ft), inches(in) {}

    void getdistance(){
      cout << "\nEnter feet:"; cin >> feet;
      cout << "Enter inches:"; cin >> inches;  }

    void showdist() {cout << feet << "\' " << inches << "\""; }

    Distance operator + (Distance) const;
};

Distance Distance::operator + (Distance d2) const {

  int f = feet + d2.feet;
  float i = inches + d2.inches;

  if(i >= 12.0) {
    i -= 12.0;
    f++;  }

  return Distance(f,i);  }

int main() {

  Distance dt1, *dt3, dt4;

  dt1.getdistance();

  Distance dt2(8,5);

  dt3 = &dt1;
  dt4 = dt1 + dt2;
  dt1 = dt4 + dt2;

  cout << "Distance1 = "; dt1.showdist(); cout << endl;
  cout << "Distance2 = "; dt2.showdist(); cout << endl;
  cout << "Distance3 = "; dt3->showdist(); cout << endl;
  cout << "Distance4 = "; dt4.showdist(); cout << endl;
}
```

Index Number | | | | | | | | |

(a). Briefly explain the difference between a `Class` and an `Object`.

[2 marks]

(b). Give an example for one (1) `Class` and one (1) `Object` from the above code segment.

Class: **Object:**

[1 marks]

(c). Provide examples for each of the following considering the above program. (Including only the signature of the constructor/the function is sufficient).

i.) Default Constructor:

ii.) Parametrized Constructor:

iii.) Member Function:

[3 marks]

Index Number | | | | | | | | |

(d). Write down the console output of the above code.
Consider the console inputs for the variables as $feet = 5$ and $inches = 2.5$.

[10 marks]

(e). Write down a Destructor function that is syntactically correct to display the below when an object is being destroyed.
`Destroy OBJECT`

[4 marks]

(f). Suppose you have implemented the Destroy Function written in (2.e above) syntactically correct. Write down the output that will be displayed on the console after implementing the Destroy function.

[5 marks]

Index Number								

3. (a). Only ONE answer is correct in the following 12 MCQs. Cross or color the **best suite answer** among the given options.

[2 x 12 = 24 marks]

i.	(A)	(B)	(C)	(D)	(E)
ii.	(A)	(B)	(C)	(D)	(E)
iii.	(A)	(B)	(C)	(D)	(E)
iv.	(A)	(B)	(C)	(D)	(E)
v.	(A)	(B)	(C)	(D)	(E)
vi.	(A)	(B)	(C)	(D)	(E)

vii.	(A)	(B)	(C)	(D)	(E)
viii.	(A)	(B)	(C)	(D)	(E)
ix.	(A)	(B)	(C)	(D)	(E)
x.	(A)	(B)	(C)	(D)	(E)
xi.	(A)	(B)	(C)	(D)	(E)
xii.	(A)	(B)	(C)	(D)	(E)

i. Consider the following three statements regarding the `Inheritance` in OOP.

I. It is a mechanism for creating new classes from existing ones.
II. It is a mechanism that allows objects to communicate with each other.
III. Train and Engine has the relationship define as Inheritance.

Which of the above statement(s) is/are TRUE?

A. I only.	B. I and II only.	C. II and III only.
D. III only.	E. I, II and III.	

ii. Which of the following is TRUE according to the following class definition?

```
class Bat : private Mammal, protected Bird {
        // body of the class
}
```

A. All public, protected and private data in `Mammal` will become private to `Bat`.
B. Both public and protected data in `Mammal` will become protected to `Bird`.
C. Both public and protected data in `Bird` will become protected to `Bat`.
D. only protected data in `Bird` will beocme protected to `Bat`.
E. `Bat` cannot access any data in `Mammal`.

Index Number | | | | | | | | |

iii. What is the output of the following program written in C++?

```
class A {
    public: A() { cout<<"A"; }
            A(int x) { cout<<x; } };

class B: public A {
    public: B(int y) { cout<<y; } };

int main() { B b(20); A a(10); }
```

A. 2010	B. 1020	C. 202010
D. A102010	E. A2010	

iv. Consider the following three statements regarding the `Diamond Problem` in OOP.

I. It occurs when two classes have a common base class.
II. It always prevents compiling your program.
III. `virtual` keyword can be used to prevent the diamond problem.

Which of the above statements is/are TRUE?

A. I only.	B. I and II only.	C. III only.
D. II and III only.	E. I, II and III.	

v. What is the output of the following program written in C++?

```
class Draw {
    public:
        int drawing(int a) {cout<<a*2;}
        void drawing(int b) {cout<<b;}
        int drawing (char c) {cout<<c;}
};

int main() {
        Draw d;
        d.drawing(5);
        d.drawing('X');
}
```

A. 105X	B. 10X	C. 5X
D. Compilation Error	E. Run time Error	

Index Number | | | | | | | | |

vi. The purpose of the `override` keyword in C++ is to,

A. define a new virtual method.
B. specify the access level of a method.
C. prevent a method from being overridden.
D. indicate the abstract method which should be overridden.
E. indicate that a derived class method is intended to override.

vii. Assume that there is a function named `run()` in a Abstract class called `Mammal`, which needs the weight of the mammal as a parameter to implement. Whoever, the implementation of the `run()` function can not be defined at the `Mammal` class since the weight of a mammal correctly defines at its sub classes. Which of the following is the correct definition of the `run()` function at the `Mammal` class in C++?

A. `virtual void run() = 0;`
B. `virtual void run(int w);`
C. `abstract void run(int w);`
D. `virtual void run(int w) = 0;`
E. `abstract void run(int w) = 0;`

viii. What is the output of the following program written in C++?

```
class Draw {
    public:
        A(int y) {}
        int func(int x, int y=0) { return x * y; }
};

int main() {
        A a(10);
        cout << a.func(10); }
```

A. 100 B. 1010 C. 0
D. Compilation Error E. Run time Error

ix. An exception in Object Oriented Programming can be defined as a/an;

A. predefined error message.
B. error that occurs during program execution.
C. type of function to trigger when an error occurs.
D. loop running till an error occurs.
E. technique to debug the code.

Index Number | | | | | | | | |

x. What is the output of the following C++ program?

```
int main() {

    try { throw 'A'; cout << "w "; }

    catch (int y) {  cout << "x "; }

    catch(...) { cout<<"y "; }

    cout << "z ";
}
```

A. w z	B. y z	C. w x z
D. w y z	E. w x y z	

xi. What is the output of the following C++ program?

```
template <typename T, typename U, typename V>
V func(T x, U y, V z) {
    return x * y;
}

int main() {
    cout << func<int, double, int>(5, 3.5, 2) << ", ";
    cout << func<double, int, char>(13, 5, 'C') << endl;
}
```

A. 17,5, C	B. 17.5, A	C. 17, C
D. 17, A	E. Compilation Error	

xii. Consider the following definition of a template.

```
template <typename T, int y>
T f(T x) { return x + y; }
```

Which of the fallowing is NOT a correct call of the function `f()`?

A. f<int, 10>(20)	B. f<double, 3>(3.5)	C. f<char, 3.4>('A')
D. f<char, 'A'>('A')	E. f<char, 'A'>(12))	

(b). Write a correct call of the function `f()` above *(in Question xii)* to return character `'A'` as the output.

f<char, 0>('A')

.....

[1 mark]

Index Number | | | | | | | | |

4. (a). Write the output of the following piece of programs written in C++.
Assume that all the other required lines of codes are in place to run programs.

i.
```
class A{
  public:
        A() {cout<<"A "; }
        ~A() {cout<<" ~A "; } };

class B: public A {
  public:
        B(int x) {cout<<x; }
        ~B() {cout<<"~B "; } };

int main( ){
  try { B b(5); A a; throw 10; }
  catch (int i) {cout<<i; }
}
```

[8 mark]

A 5 A -A B -A 10

.....

ii.
```
class A{
  public:
        A() {cout<<"A "; } };

class B: public A {
  public:
        B() {cout<<"B "; } };

class C: public B, public A {
  public:
        C() {cout<<"C "; } };

int main( ){
  try { C c; throw c; }
  catch (B b) {cout<<"D "; }
  catch (A a) {cout<<"E "; }
  catch (C c) {cout<<"F "; }
  catch (...) {cout<<"G "; }
  cout<<"I ";
}
```

[6 marks]

A B A C D I

.....

Index Number | | | | | | | | |

(b). Assume that there are three (3) overloaded functions named `add()` in `MyMaths` class. When they call it as follows, it will give the output as `6, 66, B, 9`

```
int main() {
        cout<<add(5)<<", ";
        cout<<add(1, 'A')<<", ";
        cout<<add('A', 1)<<", ";
        cout<<add(6, 3.5);
}
```

Implement the three (3) `add()` fucntons mentioned above.

1.)

2.)

3.)

[6 marks]

(c). Suppose we have a program that models a music streaming service. The program might have several classes, including a User class, a Playlist class, and a Song class. Each User object would have a list of Play list objects that they have created, and each Playlist object would have a list of Song objects that are included in the playlist. One Song object could be included in multiple playlists. Finally, the User class might inherit properties and methods from a Person class, which could include things like a name, an email address, and a date of birth. Identify the relationships between following classes according to the above scenario.

1.) User and Playlist: ..... User and Playlist: One-to-Many (Aggregation)

2.) Song and Playlist: ..... Song and Playlist: Many-to-Many (Association)

3.) User and Person: ..... User and Person: Inheritance

[5 marks]

****