National University of Computer & Emerging Sciences, Karachi
Spring-2022 School of Computing (BSCS, BSCY)
Final Examination
1st June 2022, 08:30 am – 11:30 am

Course Code: CS1004	Course Name: Object Oriented Programming
Instructor Name: Dr. Farooque Hassan Kumbhar, Mr. Zain ul Hassan, Ms. Farah Sadia, Ms. Nida Munawar, Ms. Abeer Gauher, Mr. Basit Ali	
Student Roll No:	Section No:

Instructions:

- Return the question paper and make sure to keep it inside your answer sheet.
- Read questions completely before answering. There are **7 questions, 4 sides on 2 pages**.
- In case of any ambiguity, you may make assumption. But your assumption should not contradict any statement in the question paper.
- You are **not allowed to write** anything on the question paper (except your ID and section).

Time: 180 minutes. **Max Marks**: 53 Marks

Q 1. **[20 min, 5 Marks (1 each), CLO #1]** Write short answers to the following questions:

a. Consider the following four calls to a global template function:
i) *func (2, "hello", 3);* **ii)** *func ('x', 5.4, 'y');* **iii)** *func (3.1, "oop", 6.2);* **iv)** *func (1, 2, 3);*
Write the definition of this template function (only its skeleton) that uses minimum possible number of generic types such that these calls can be successfully made.
Answer: Notice that the first and third arguments are always of similar types, hence based on the given examples the template function should have a minimum of two generic types and will look like:
template<class T1, class T2> void func(T1 a, T2 b, T1 c) { // code of function }

b. Assume there is a class that is inherited from two parent classes (multiple inheritance). Is it possible for Diamond Problem to occur even if none of parent or child classes contain any functions at all and only have variables? Justify your answer using an example.
Answer: Yes, it is possible for Diamond Problem to arise in this case as the attributes can also be inherited and overridden by multiple child classes.
Example:

```
class A
{ int var;
};
class B: public A {
int var;
};
class C: public A {
int var;
};
class D: public B, public C
{
// using object of class D to use var will cause Diamond Problem
};
```

c. What is the difference between virtual function and pure virtual function?
Answer: A virtual function have an implementation and may or may not be overridden by the child class, whereas a pure virtual function does not an implementation and must always be overridden by the child class.

d. Briefly explain with examples, IS-A and HAS-A relationships.
Answer: An IS-A relationship expresses extension or sub-types. In OOP, it is represented through classes. A HAS-A relationship expresses ownership or usage. In OOP, it is represented when object or one class is used in another class.

e. Should a default catch block always be the last of the catch blocks if there are more than one blocks present? Justify your answer.
Answer: Yes. A default catch block must always be the last catch block. Since it can catch any type of exception object, if there are any more catch blocks after the default catch block they will never be reached.

Q 2. Fill in the blanks of the following questions. **[15 min, 5 Marks (0.5 each), CLO #1]**

a. If we have an object from ofstream class, then the default mode of opening the file is _____.
b. _____ creates different versions of a function at runtime.
c. _____ returns true if a file open for reading has reached the end.
d. _____ feature can be implemented using encapsulation.
e. The block that is used to check for errors is called as the _____ block
f. Class template can be created using _____ syntax.
g. Member functions having the same name in base and derived class is called as _____.
h. _____keyword is used to handle the exception.
i. The following expression: (3 > 6) && (7 > 4) will return _____.
j. A function that is called automatically each time an object is destroyed is a _____.

ANSWERS:

a. ios::out
b. Template functions
c. eof()
d. Abstraction
e. try
f. template<class T>class classname
g. Overriding
h. Catch
i. False
j. Destructor

Q 3. Review following given codes and identify any errors that might occur. If there are no errors, then write output of the programs. **[15 min, 5 Marks (1 each), CLO #2]**

a)
```
template<class T, class U, class V=double>
class A  {
   T x;
   U y;
   V z;
   static int count;
};

int main()
{
  A<int, int> a;
  A<double, double> b;
  cout << sizeof(a) << endl;
  cout << sizeof(b) << endl;
  return 0;
}
```
ANSWER:
16
24

b)
```
class Base{
public:
        virtual void display()=0;
};
class Child: public Base{
public:
        virtual void display(){
        cout<<"Never Give up!!!"<<endl;
        };
};
int main(){
Child obj;
cout<<sizeof(obj)<<endl;
return 0;
}
```
ANSWER:
8

c)
```
class A{
public:
        void square (int *x){
                *x = (*x)++ * (*x);
        }
        void square (int *x, int *y){
                *x = (*x) * --(*y);
        }
};
int main(){
A obj;
int number = 10;
obj.square(&number, &number);
cout <<number;
return 0;
}
```
ANSWER:
90

d)
```
int main() {
  int var = 0;
  try {
        try {
        throw var;
        }
        catch (int ex) {
        ex+=10;
        cout << "Error handling :: Val :" << var<<" Ex :
"<<ex<<endl;
        throw;
        }
}
        catch (...) {
        cout << "All Exception Catch: val :"<< var<<"
Ex : "<<ex<<endl;
   }
  return 0;
}
```
ANSWER:
[Error] 'ex' was not declared in the following score
cout << "All Exception Catch: val :"<< var<<" Ex : "<<ex<<endl;
After correction:
cout << "All Exception Catch: val :"<< var<<endl;

After correction output will be:
Error handling :: Val :0 Ex : 10
All Exception Catch: val :0

e)
```
class A{
        int x;
public:
        A(int i) { x = i; }
        void print() { cout << x; }
};
class B: virtual public A{
public:
        B():A(10) { }
};
class C: virtual public A{
public:
        C():A(10) { }
};
```

class D: public B, public C {
};
int main(){
D d;
d.print();
return 0;
}
[Error] no matching function for call to 'A::A()'	

Q 4. Consider the given classes Sport and Tournament. **[40 min, 12 Marks (3 each), CLO #3]**

```
class Sport
{       string current_champ;
        int start_year;
        string headquarter_location;
};
class Tournament
{
        Sport * sport;
};
```

a. Write a default constructor for Tournament class that initializes Sport class reference. Also, write an overloaded constructor for Tournament class that initializes Sport class reference and receives the attributes for Sport in parameters.
b. Write a function Begin_Tournament in class Tournament. This function should receive Sport as argument and print "Tournament begins" if the headquarters of the given sport is either Karachi or Toronto.
c. Create a new class World Cup and inherit it from Tournament. Override the function Begin_Tournament in this derived class. The derived class function should print "World Cup begins" if the starting year of given sport is after 1950 and if the headquarters of the sport is neither Karachi nor Toronto.
d. Create a global generic function name PrintIt that takes any attribute of Sport as input and display its value.

a. Write a default constructor for Tournament class that initializes Sport class reference. Also, write an overloaded constructor for Tournament class that initializes Sport class reference and receives the attributes for Sport in parameters.

Answer:

```
// other code
public:
// Default constructor
Tournament( ) {
sport = new Sport; // assume implementation of default constructor
}
// Overloaded constructor
Tournament(string champ, int year, string hq) {
sport = new Sport(champ, year, hq);
// assume implementation of parameterized constructor in Sport class
}
```

b. Write a function Begin_Tournament in class Tournament. This function should receive Sport as argument and print "Tournament begins" if the headquarters of the given sport is either Karachi or Toronto.

Answer:

```
class Tournament {
// other code
public:
void Begin_Tournament(Sport s)
{
```

```
    if(s.getHeadquarter( ) == “Karachi” || s.getHeadquarter( ) == “Toronto”)
    {
            cout << “Tournament begins”;
    }
}
```

c. Create a new class World Cup and inherit it from Tournament. Override the function Begin_Tournament in this derived class. The derived class function should print "World Cup begins" if the starting year of given sport is after 1950 and if the headquarters of the sport is neither Karachi nor Toronto.

Answer:

```
// child class of Tournament
class WorldCup : public Tournament {
void Begin_Tournament(Sport s) {
if((s.getStartDate( ) > 1950) && (!(s.getHeadquarter( ) == “Karachi”)) &&
((!s.getHeadquarter( ) == “Karachi”)))
    {
            cout << World Cup begins”;
    }
}
```

d. Create a global generic function name PrintIt that takes any attribute of Sport as input and display its value.

Answer:

```
template<class T> void PrintIt(T var)
{
    // “T var” can be any given variable of Sport class
    cout << var;
}
```

Q 5. [30 min, 9 Marks (3 each), CLO #5] Suppose you are running a jumping castle playland and for safety reasons want to implement a rule that at most 10 people can be on the jumping castle at any one time. People are allowed to go on for as long as they wish. In order to make certain there are not too many people on the castle you station an employee at the entrance/exit. This employee will count people as they enter and count them as they exit. If the jumping castle is full then a person wishing to enter will need to wait until someone first exits.

a. Write a program that displays the current number of people on the jumping castle, have an "Enter" function that increments the people count, an "Exit" function that decrements the people count. The program should repeat indefinitely and every time asks the employee to either enter or exit people.

b. The people count should not be permitted to go beyond 10 when employee enters 11th person it should throw an exception that prints a string message "Count Overflow"

c. Also, there shouldn't be a negative number of people on the castle! When employee decrements at 0th position it should throw an exception that prints a string message "Count Underflow").

```
#include <iostream>
#include <string>
using namespace std;
static int current = 0;
void enter() {
        int size;
  cout << "enter size to increment" << endl;
  cin >> size;
for (int i = 1 ; i <= size ; i++){
 current++;
                                        if(current>10) {
                                                current--;
                                                throw "Count overflow";}}}
void exit() {
int size;
  cout << "enter size to decrement" << endl;
  cin >> size;
for (int i = current ; i > size ; i--){
 current--;
                                        if(current<0) {
                                                current++;
                                                throw "Count overflow";}}}

int main() {
  try {
        enter();                }
                                        catch(...) {
                                        cout << "Count overflow" << endl;
                                        cout << "if value more then 10  current people count set to 10 : "
<< current << endl ;
                                }


  try {
        exit();                 }
```

```
                                        catch(...) {
                                        cout << "Count underflow"<< endl;
                                        cout << "if value in negative current people count set to 0 : " <<
current << endl ;
                                        }
                                        cout << "current people count : " << current << endl ;
  return 0;}
```

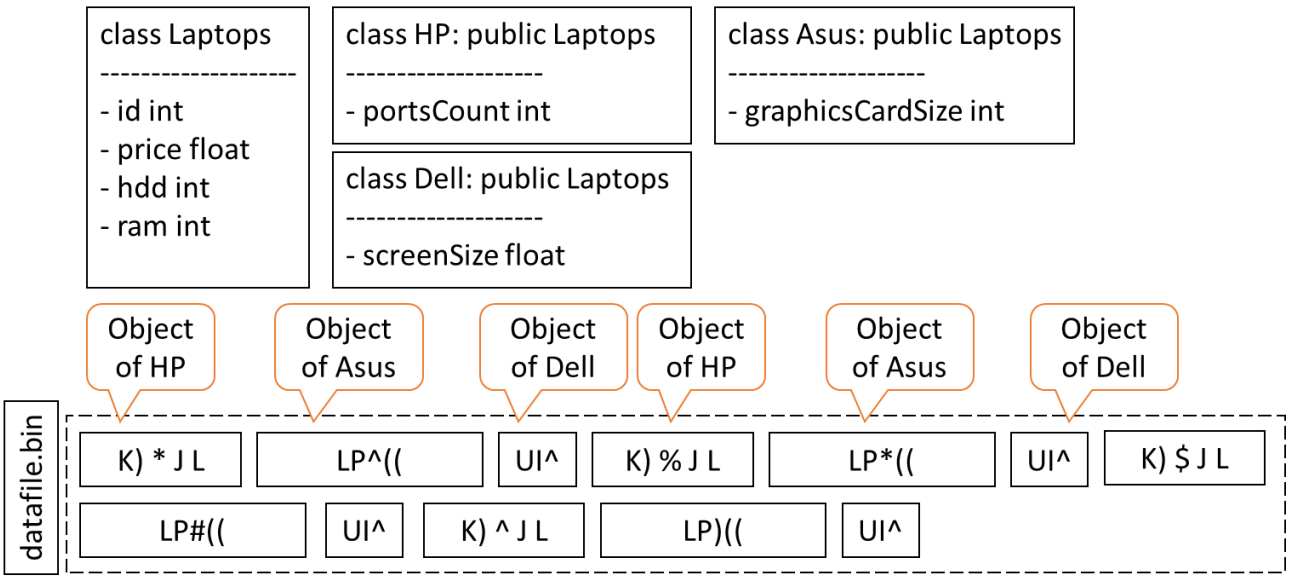
Q 6. [20 min, 9 Marks (3 each), CLO #3] Declare a class FILE as an abstract class. The class contains data members like size, location, created date and modified date, and functions like open() and print(). Open() is a pure virtual function and print() is a virtual function.

a. Derive three classes (PDF file, ASCII File, and PS File) from FILE class, such that there is slight difference between the implementation of print() function in all derived classes.
b. Create 3 File class pointers in main, and store the reference objects of each subclass in them and call their respective print() functions polymorphically. Delete all allocated objects memory before program exits.
c. Develop a global function that can be used to identify two objects of FILE type and finds duplicates. The function should be able to access class members but make sure that function must not be able to change any value. The new function compares size and location of two files and if size of the objects matches and location is same, the function returns true otherwise, false. The function signature is given below for your reference.
bool operator * (File file_object_1, File file_object_1).

```
class File{
        protected:
        int size;
        string location;
        string created_date;
        string modified_date;
        public:
                File(){}
                File(string l, int s, string cd, string md):location(l),size(s),  created_date(cd), modified_date(md){}
                void Open(){
                        cout<<"Open File"<<endl;
                }
                virtual void Print(){
                }
                virtual ~File(){cout<<"File is deleted successfully"<<endl;}
};
class PDF:public File{
        public:
                PDF(){}
                PDF(string l, int s, string cd, string md):File(l, s, cd, md){}
                void Print(){
                        cout<<"Print PDF "<<endl;
                }
                void properties(){
```

```
                cout<<"Path : "<<location<<"\nSize : "<<size<<"\nCreated date :
"<<created_date<<"\nModified date : "<<modified_date<<endl;

        }

        ~PDF(){cout<<"PDF file is deleted successfully"<<endl;}

};

class ASCII: public File{

    public:

        ASCII(){}

        ASCII(string l, int s, string cd, string md):File(l,s,  cd, md){}

        void Print(){

                cout<<"Print ASCII "<<endl;

        }

        void properties(){

                cout<<"Path : "<<location<<"\nSize : "<<size<<"\nCreated date :
"<<created_date<<"\nModified date : "<<modified_date<<endl;

        }

        ~ASCII(){cout<<"ASCII file is deleted successfully"<<endl;}

};

class PS:public File{

    public:

        PS(){}

        PS(string l, int s, string cd, string md):File(l,s, cd, md){}

        void Print(){

                cout<<"Print PS "<<endl;

        }

        void properties(){

                cout<<"Path : "<<location<<"\nSize : "<<size<<"\nCreated date :
"<<created_date<<"\nModified date : "<<modified_date<<endl;

        }

        ~PS(){cout<<"PS file is deleted successfully"<<endl;}

};

int main(){

    File *f1=new PDF();

    File *f2=new ASCII();

    File *f3=new PS();

    f1->Print();
```

```
        f2->Print();
        f3->Print();
        delete f1;
        delete f2;
        delete f3;
}
```



Q 7. [20 min, 8 Marks (4 each), CLO #4] Let's assume four classes as given in the above picture. You are provided a "datafile.bin" file where 100's of objects are stored for an electronic shop. The sequence of the objects in "datafile.bin" is also described in the above given diagram. Write a program where:

a. Read the "datafile.bin" to extract all objects, and print their details.
b. Calculate total price of all HP, ASUS, and DELL Laptops, separately. Create another file "report.txt" where store total sale amount with respect to each type of the laptop. The report.txt file should look like as given in the adjacent figure.

Report.txt

```
ELECTRONICS SHOP:
-------------------------
HP: 9250000 PKR
DELL: 679200 PKR
ASUS: 2990920 PKR
-------------------------
Total: 12920120 PKR
```

ANSWER:

a)

```
ifstream input_ob;
  input_ob.open("datafile.bin");

  while(input_ob)
  {
    HP hpOb;
    Asus asusOb;
    Dell dellOb;

    input_ob.read((char*)&hpOb, sizeof(hpOb));
    input_ob.read((char*)&asusOb, sizeof(asusOb));
    input_ob.read((char*)&dellOb, sizeof(dellOb));

    //Assuming that display function is outlined by the students
    //Otherwise, they will have to print all members in cout statement.
    hpOb.display();
    asusOb.display();
```

```
        dellOb.display();

    }
    input_ob.close();
```

b)

```
   ifstream input_ob;
    input_ob.open("datafile.bin");
    int totalHPSale = 0, totalAsusSale = 0, totalDellSale = 0;
    while(input_ob)
    {
        HP hpOb;
        Asus asusOb;
        Dell dellOb;

        input_ob.read((char*)&hpOb, sizeof(hpOb));
        input_ob.read((char*)&asusOb, sizeof(asusOb));
        input_ob.read((char*)&dellOb, sizeof(dellOb));


        totalHPSale = totalHPSale + hpOb.price;
        totalAsusSale = totalAsusSale + asusOb.price;
        totalDellSale = totalDellSale + dellOb.price;

    }
    input_ob.close();

    ofstream output_ob;
    output_ob.open("report.txt");
    output_ob << "ELECTRONICS SHOP\n"<<"---------------\n";
    output_ob << "HP:" << totalHPSale << "PKR\n";
    output_ob << "Dell:" << totalDellSale << "PKR\n";
    output_ob << "Asus:" << totalAsusSale << "PKR\n";
    output_ob << "-------------------------\n";
    output_ob << "Total:" << (totalHPSale + totalDellSale + totalAsusSale) << "PKR\n";
```

BEST OF LUCK!