CS1004 Object Oriented Programming

Wednesday, May 31, 2023

Course Instructor

Dr. Bilal Khan, Dr. Imran Babar, Dr. Khalid Hussain, Rizwan ul Haq, Usman Ghous, Saud Arshad

Serial No:

Final Exam-Objective

Total Time: 40 M

Total Marks: 35

_____
Signature of Invigilator

Roll No _____ Section _____ Signature _____

DO NOT OPEN THE QUESTION BOOK OR START UNTIL INSTRUCTED.

Instructions:

1. Verify at the start of the exam that you have a total of two (2) questions printed on Nine (09) pages including this title page.
2. Attempt all questions on the question-book and in the given order.
3. The exam is closed books, closed notes. Please see that the area in your threshold is free of any material classified as 'useful in the paper' or else there may a charge of cheating.
4. Read the questions carefully for clarity of context and understanding of meaning and make assumptions wherever required, for neither the invigilator will address your queries, nor the teacher/examiner will come to the examination hall for any assistance.
5. Fit in all your answers in the provided space.
6. Use only your own stationery and calculator. **Calculator is not allowed**.
7. Use only permanent ink-pens. Only the questions attempted with permanent ink-pens will be considered. Any part of paper done in lead pencil cannot be claimed for checking/rechecking.
8. Subjective part can be given to the student upon submitting the Objective part at any time.

<Objective Part>

	Q-1	Q-2	Total
Total Marks	20	15	**35**
Marks Obtained			

Vetted By: _____ **Vetter Signature:** _____

University Answer Sheet Required: **No** ■ **Yes** □

Q1.	20

Fill the appropriate box with best answer for the MCQs provided.

Q. #	Answer			
1.	A		C	D
2.	A	B		D
3.		B	C	D
4.	A	B	C	
5.	A	B		D
6.	A		C	D
7.	A	B		D
8.	A	B	C	
9.	A	B		D
10.	A	B		D
11.	A	B	C	
12.		B	C	D
13.		B	C	D
14.		B	C	D
15.	A	B	C	
16.	A		C	D
17.		B	C	D
18.	A	B	C	
19.		B	C	D
20.		B	C	

1. The OOPs concept in C++, exposing only necessary information to users or clients is known as
   A. Inheritance
   B. Abstraction
   C. Encapsulation
   D. Polymorphism

2. What will be the output when an instance of class D is created:
```
class A {public: A() { cout << "A"; };};
class B : public A { public: B() { cout << "B"; } };
class C : public A { public: C() { cout << "C"; } };
class D : public B, public C { public: D() { cout << "D"; } };
```
   A. ABCD
   B. DCBA
   C. ABACD
   D. DCABA

3. What will be the output when an instance of class D is created:
```
class A {public: A() { cout << "A"; };};
class B : virtual public A { public: B() { cout << "B"; } };
class C : virtual public A { public: C() { cout << "C"; } };
class D : public B, public C { public: D() { cout << "D"; } };
```
   A. ABCD
   B. DCBA
   C. ABACD
   D. DCABA

4. What will be the output when an instance of class D is destroyed:
```
class A {public: ~A() { cout << "A"; };};
class B : public A { public: ~B() { cout << "B"; } };
class C : public A { public: ~C() { cout << "C"; } };
class D : public B, public C { public: ~D() { cout << "D"; } };
```
   A. ABCD
   B. DCBA
   C. ABACD
   D. DCABA

5. Constructor of a class cannot be _____
   A. virtual
   B. private
   C. friend
   D. None of the above is a correct answer

6. We cannot have overloaded
   A. Operators
   B. Destructor

C. Constructor
D. Function

7. A pointer to the base class can hold address of
   A. Only base class instance
   B. Only derived class instance
   C. Base class instance as well as derived class instance
   D. None of the above

8. How can a static member function be called in main function?
   A. Using dot operator
   B. Using arrow operator
   C. Using dot or arrow operator
   D. Using dot, arrow or using scope resolution operator with class name

9. Which of the following operator cannot be overloaded?
   A. []
   B. new
   C. ::
   D. +=

10. If same function is called from objects of several different classes and all of those can respond in a different way, what is this feature called?
    A. Inheritance
    B. Overloading
    C. Polymorphism
    D. Overriding

11. Which among the following best defines static data members?
    A. Data which is common to all the classes
    B. Data which is common to a specific method
    C. Data which is allocated for each object separately
    D. Data which is common to all the objects of a class

12. What is the memory address of this pointer
    A. It does not have its own memory address as it is register variable
    B. The address is the start of the object as it is part of object
    C. Both A or B depending upon situation
    D. None of the above

13. A constant member function can access
    A. All members of the class
    B. Only static members of the class
    C. Only const members of the class
    D. Only Static and const members of the class

14. Where does keyword 'friend' should be placed?
    A. function declaration
    B. function definition
    C. main function
    D. None of the mentioned

15. Which statement is true, if we execute the given code

```
int main() {
    int *iPtr;
    float b = 5;
    iPtr = &b;
    cout << *iPtr << endl;
}
```

    A. It will print 5
    B. It will print address of b
    C. It will be a runtime error
    D. It will be a compiler error

16. What will be the result of Line 8

```
class Class1 {      //Line 1
    int x ;         //Line 2
};
class Class2 {      //Line 3
    int y = 15;     //Line 4
};
int main()          //Line 5
{
    Class1 c1;      //Line 6
    Class2 c2;      //Line 7
    c1 = c2;        //Line 8
}
```

    A. Will do nothing
    B. Will generate a syntax error
    C. Will generate a runtime error
    D. Will set the value of x in c1 to 15

17. A public member function foo() can be accessed by its pointer ptr in the following manner
    A. ptr->foo() only
    B. *ptr.foo() only
    C. *ptr.foo() and ptr->foo()
    D. ptr*.foo() and ptr->foo()

18. An array of the objects of the following class can be created as

```
class myClass {
    int x;
    int y;
public:
    myClass(int a, int b) {
            x = a; y = b;
    }
};
```

A. myClass *obj = new myClass[2] (5, 5);
B. myClass *obj = new myClass[2] {5, 5};
C. myClass *obj = new myClass[2] { (5, 5), (15,15)};
D. myClass *obj = new myClass[2] { {5, 5}, { 15,15 }};

19. The relationship between Class1 and Class2 in the underneath given code is

```
class Class1 {
    int a;
public:
    Class1() :a(0) {}
};
class Class2 {
    int b;
    Class1 *objClass1;
public:
    Class2(){
            objClass1 = new Class1[5];
    }
    ~Class2() {
            delete[] objClass1;
    }
};
```

A. Composition
B. Aggregation
C. Inheritance
D. Polymorphism

20. correct syntax for a function pointer for a function with prototype int foo(float a, int b); is

A. int (*fPtr)(float, int)
B. int *fPtr(float, int)
C. int *fPtr(float a, int b)
D. int (*fPtr)(float a, int b)

Q2.	3*5 = 15

Write down the output of following given program in the second column. **If the code contains some error or missing statement, write down the corrected line of code and the output after correction in the space provided.**

Sr.	Code	Output

a.	`class A {`
	`public:`
	`  A() { cout << 'A'; }`
	`  ~A() { cout << "A1\n"; }`
	`};`
	`class B:public A {`
	`public:`
	`  B() { cout << 'B'; }`
	`  ~B() { cout << "B1\n"; }`
	`};`
	`class C:public A`
	`{`
	`public:`
	`  C() { cout << 'C'; }`
	`  ~C() { cout << "C1\n"; }`
	
	`};`
	`int main()`
	`{`
	`  A* ptrA[4];`
	`  for (int i = 0; i < 4; ++i)`
	`  {`
	`  if (i % 2)`
	`  ptrA[i] = new B;`
	`  else`
	`  ptrA[i] = new C;`
	`  cout << endl;`
	`  }`
	`  for (int i = 0; i < 4; ++i)`
	`  {`
	`  delete ptrA[i];`
	`  }`
	`  return 0;`
	`}`
		Output:
		AC
		AB
		AC
		AB
		A1
		A1
		A1
		A1

b.

```
class A {
public:
	~A() { cout << 'A' << endl; }
};
class B :private A {
public:
	~B() { cout << 'B' << endl; }
};
int main()
{
	B b;
	A a;
	{
		B * Bptr;
		A * Aptr = &a;
	}
	return 0;
}
```

Output:

```
A
B
A
```

c.

```
class A {
	int a;
public:
	A():a(10) {}
	int getVal() { return a; }
	~A() { cout << 'A'; }
};
class B :public A {
	int b;
public:
	B() :b(15) {}
	int getVal() { return b; }
	~B() { cout << 'B'; }
};
class C :public A {
	int c;
public:
	C():c(20) {}
	~C() { cout << 'C'; }
	virtual int getVal() { return c; }
};
int main()
{
	A* ptrA = new B;
	cout << ptrA->getVal() << endl;;
	ptrA = new C;
	cout << ptrA->getVal() << endl;;
}
```

Output:

```
10
10
```

d.

```
int foo(int *iPtr, int i)
{
	if (i < 0)
		return 0;
	else if (i < 4)
		return iPtr[0] + *(iPtr - 2);
	else
		return *iPtr + foo(iPtr-1, i-1);
}
int main()
{
	int arr[] = { 5,3,2,4,5,6 };
	cout << foo(arr + 5, 5);
}
```

Output:

```
18
```

e.

```
template<class T>
void PrintSum(T a, T b)
{
	cout << a + b;
}

template<class T, class U>
void PrintSum(T a, U b)
{
	cout << a + b;
}

int main()
{
	int a = 2, b = 7;
	double c = 1.25, d = 2.25;
	PrintSum(a, b);
	cout << endl;
	PrintSum(c, d);
	cout << endl;
	PrintSum(a, c);
	cout << endl;
	PrintSum(c, a);
	cout << endl;
	PrintSum(10, 9);
}
```

How many Instances of the function will be created: ___2____

Output:

```
9
3.5
3.25
3.25
19
```