CP Test-6

Computer Programming Language - Test 6

Email *

jay.chachapara.ce@gmail.com

NAME OF THE STUDENT (Write in CAPITAL) *

JAY CHACHAPARA

Enrollment no: (Write completely and in CAPITAL. Ex: BT19CIV001) *

MT21MCS013

Year:
*

- (●) 1st Year
- ( ) 2nd Year
- ( ) 3rd Year

Your Academic Programme *

- ( ) B.Tech
- (•) M.Tech
- ( ) M.Sc
- ( ) B.Arch

Branch
Specialization *

M.Tech CSE

Your Training
BATCH *

- ( ) Batch 1
- (•) Batch 2
- ( ) Batch 3

Technical Section

Which one is not a correct variable type in C++?

- ○ float
- ◉ real
- ○ int
- ○ double

Which operation is used as Logical 'AND'

- ○ Operator-&
- ○ Operator-||
- ◉ Operator-&&
- ○ Operator +

Correct answer

◉ Operator-&

An expression A.B in C++ means ____

- ○ A is member of object B
- ◉ B is member of Object A
- ○ Product of A and B
- ○ None of these

A pointer pointing to a variable that is not initialized is called ____

- ○ Void Pointer
- ○ Null Pointer
- ○ Empty Pointer
- ◉ Wild Pointer

Correct answer

- ◉ Null Pointer

Default constructor has ____ arguments.

- ◉ No argument
- ○ One Argument
- ○ Two Argument
- ○ None of these

A class whos objects can not be created is known as _____

- ○ Absurd Class
- ○ Empty Class
- ○ Super Class
- ◉ Abstract Class

In CPP, members of a class are _____ by default.

- ○ Public
- ◉ Private
- ○ Protected
- ○ Static

In C++ Program, inline fuctions are expanded during ____

- ○ Run Time
- ◉ Compile Time
- ○ Debug Time
- ○ Coding Time

Correct answer

- ◉ Run Time

Keywords are also called as-

- ○ Preprocessors
- ◉ Reserved words
- ○ Punctuation marks
- ○ Operators

Overflow is one kind of-

- ○ Syntax error
- ○ Logical error
- ◉ Runtime error
- ○ None of them

In C++, the name assigned by user for a program element, such as variable, function or namespaces is called-

- ○ Operators
- ◉ Identifiers
- ○ Literals
- ○ All of them

The string that contains zero character is called-

- ◉ Empty String
- ○ Idle String
- ○ Vacant String
- ○ None of them

Sequence of character delimited by quotation marks is called-

- ○ Array
- ◉ String Literals
- ○ Linked List
- ○ None of them

C++ comprises a confirmation of both high level and low level language features, that's why it is called as-

- ○ Object Oriented
- ◉ Intermediate level language
- ○ Compiled Language
- ○ None of them

A variable is symbol that represents-

- ○ A number
- ○ A string
- ◉ A storage location is computer's memory
- ○ Nothing

What happens when we try to compile the class definition in following code snippet?

```
class Birds {};
class Peacock : protected Birds {};
```

- ○ It will not compile because class body of Birds is not defined.
- ○ It will not compile because class body of Peacock is not defined.
- ○ It will not compile because a class cannot be protectedly inherited from other class.
- ◉ It will compile succesfully.

Which of the following statements is incorrect?

- ○ Friend keyword can be used in the class to allow access to another class.
- ○ Friend keyword can be used for a function in the public section of a class.
- ○ Friend keyword can be used for a function in the private section of a class.
- ◉ Friend keyword can be used on main().

Which of the following statements is correct when a class is inherited publicly?

- ○ Public members of the base class become protected members of derived class.
- ○ Public members of the base class become private members of derived class.
- ○ Private members of the base class become protected members of derived class.
- ◉ Public members of the base class become public members of derived class.

Which of the following statements about virtual base classes is correct?

- ( ) t is used to provide multiple inheritance.
- (•) It is used to avoid multiple copies of base class in derived class.
- ( ) It is used to allow multiple copies of base class in a derived class.
- ( ) It allows private members of the base class to be inherited in the derived class.

Can a class have virtual destructor?

- (•) Yes
- ( ) No
- ( ) Maybe

Copy constructor must receive its arguments by _____ .

- ( ) either pass-by-value or pass-by-reference
- ( ) only pass-by-value
- (•) only pass-by-reference
- ( ) only pass by address

Predict the output-

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

class Base
{
public:
    virtual void show() = 0;
};

int main(void)
{
    Base b;
    Base *bp;
    return 0;
}
```

- ( ) There are compiler errors in lines "Base b;" and "Base bp;"
- (•) There is compiler error in line "Base b;"
- ( ) There is compiler error in line "Base bp;"
- ( ) No compiler Error

Predict the output-

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

class Base
{
public:
    virtual void show() = 0;
};

class Derived: public Base
{
public:
    void show() { cout<<"In Derived n"; }
};

int main(void)
{
    Derived d;
    Base &br = d;
    br.show();
    return 0;
}
```

- ( ) Compiler Error in line "Base &br = d;"
- ( ) Empty Output
- (•) In Derived

Can a constructor be virtual? Will the following program compile?

```
#include <iostream>
using namespace std;
class Base {
public:
  virtual Base() {}
};
int main() {
   return 0;
}
```

- ( ) Yes
- (•) No
- ( ) Maybe

Can static functions be virtual? Will the following program compile?

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

class Test
{
   public:
      virtual static void fun()  { }
};
```

- ○ Yes
- ◉ No
- ○ Maybe

Predict the output of following C++ program. Assume that there is no alignment and a typical implementation of virtual functions is done by the compiler.

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

class A
{
public:
    virtual void fun();
};

class B
{
public:
   void fun();
};

int main()
{
    int a = sizeof(A), b = sizeof(B);
    if (a == b) cout << "a == b";
    else if (a > b) cout << "a > b";
    else cout << "a < b";
    return 0;
}
```

- (●) a > b
- ( ) a == b
- ( ) a < b
- ( ) Compiler Error

Predict the output-

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

class A
{
public:
    virtual void fun() { cout << "A::fun() "; }
};

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

class C: public B
{
public:
   void fun() { cout << "C::fun() "; }
};

int main()
{
    B *bp = new C;
    bp->fun();
    return 0;
}
```

- ○ A::fun()
- ○ B::fun()
- ◉ C::fun()

Predict the output

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

class Base
{
public:
    virtual void show() { cout<<" In Base n"; }
};

class Derived: public Base
{
public:
    void show() { cout<<"In Derived n"; }
};

int main(void)
{
    Base *bp = new Derived;
    bp->Base::show();  // Note the use of scope resolution here
    return 0;
}
```

- (●) In Base n
- ( ) In Derived n
- ( ) Compiler Error
- ( ) Runtime Error

Which of the following statements is correct? 1.Once a reference variable has been defined to refer to a particular variable it can refer to any other variable. 2.A reference is not a constant pointer.

- ( ) Only 1 is correct.
- ( ) Only 2 is correct.
- ( ) Both 1 and 2 are correct.
- (●) Both 1 and 2 are incorrect.

Which of the following statement is correct about the references?

- ○ A reference must always be initialized within functions.
- ○ A reference must always be initialized outside all functions.
- ◉ A reference must always be initialized.
- ○ Both A and C.

Which of the following operators cannot be overloaded

- ○ . (Member Access or Dot operator)
- ○ ?: (Ternary or Conditional Operator)
- ○ :: (Scope Resolution Operator)
- ○ .* (Pointer-to-member Operator)
- ◉ All of the above

Which of the following operators are overloaded by default by the compiler in every user defined classes even if user has not written? 1) Comparison Operator (==) 2) Assignment Operator (=)

- ○ Both 1 and 2
- ○ Only 1
- ◉ Only 2
- ○ None of the two

Predict the output

```
#include<iostream>
using namespace std;
class A
{
    int i;
public:
    A(int ii = 0) : i(ii) {}
    void show() {  cout << i << endl;  }
};

class B
{
    int x;
public:
    B(int xx) : x(xx) {}
    operator A() const {  return A(x); }
};

void g(A a)
{
    a.show();
}

int main()
{
    B b(10);
    g(b);
    g(20);
    return 0;
}
```

- ( ) Compiler Error
- (•) 10 20
- ( ) 20 20
- ( ) 10 10

Predict the output

```
#include <iostream>
using namespace std;
class Test2
{
    int y;
};

class Test
{
    int x;
    Test2 t2;
public:
    operator Test2 ()  { return t2; }
    operator int () { return x; }
};

void fun ( int x) { cout << "fun(int) called"; }
void fun ( Test2 t ) { cout << "fun(Test 2) called"; }

int main()
{
    Test t;
    fun(t);
    return 0;
}
```

- ○ fun(int) called
- ○ fun(Test 2) called
- ◉ Compiler Error: Ambiguous call to fun()

Predict the output

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

class Point {
private:
  int x, y;
public:
  Point() : x(0), y(0) { }
  Point& operator()(int dx, int dy);
  void show() {cout << "x = " << x << ", y = " << y; }
};

Point& Point::operator()(int dx, int dy)
{
    x = dx;
    y = dy;
    return *this;
}

int main()
{
  Point pt;
  pt(3, 2);
  pt.show();
  return 0;
}
```

- (●) x = 3, y = 2
- ( ) Compiler Error
- ( ) x = 2, y = 3

Which of the following is true about this pointer?

- ( ) It is passed as a hidden argument to all function calls
- (●) It is passed as a hidden argument to all non-static function calls
- ( ) It is passed as a hidden argument to all static functions
- ( ) None of the above

What is the use of this pointer?

- [ ] When local variable's name is same as member's name, we can access member using this pointer.
- [ ] To return reference to the calling object
- [ ] Can be used for chained function calls on an object
- [x] All of the above

Predict the output

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

class Test
{
private:
  int x;
public:
  Test(int x = 0) { this->x = x; }
  void change(Test *t) { this = t; }
  void print() { cout << "x = " << x << endl; }
};

int main()
{
  Test obj(5);
  Test *ptr = new Test (10);
  obj.change(ptr);
  obj.print();
  return 0;
}
```

- [ ] x = 5
- [ ] x = 10
- [x] Compiler Error
- [ ] Runtime Error

Which header file is required to use file I/O operations?

- <ifstream>
- <ostream>
- <fstream>
- <iostream>

By default, all the files in C++ are opened in _____ mode.

- Text
- Binary
- ISCII
- VTC

Which of the following is the default mode of the opening using the ofstream class?

- ios::in
- ios::out
- ios::app
- ios::trunc

Which of the following is the default mode of the opening using the fstream class?

- ( ) ios::in
- ( ) ios::out
- (•) ios::in|ios::out
- ( ) ios::trunc

What is the return type is_open() method?

- ( ) int
- ( ) char
- (•) bool
- ( ) float

Predict the output of following C++ program.

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

class Test
{
    static int x;
public:
    Test() { x++; }
    static int getX() {return x;}
};

int Test::x = 0;

int main()
{
    cout << Test::getX() << " ";
    Test t[5];
    cout << Test::getX();
}
```

- [ ] 0 0
- [ ] 5 5
- [x] 0 5
- [ ] Compiler Error

Which of the following is true?

- [ ] Static methods cannot be overloaded.
- [ ] Static data members can only be accessed by static methods.
- [ ] Non-static data members can be accessed by static methods.
- [x] Static methods can only access static members (data and methods)

Predict the output of following C++ program.

```
#include <iostream>
#include <string>
using namespace std;
class complex
{

    int i;
    int j;
        public:
    complex(int a, int b)
        {
        i = a;
        j = b;
    }

    complex operator+(complex c)
        {
        complex temp;
        temp.i = this->i + c.i;
        temp.j = this->j + c.j;
        return temp;
    }

    void show(){
        cout<<"Complex Number: "<<i<<" + i"<<j<<endl;
    }
};

int main(int argc, char const *argv[])
{
    complex c1(1,2);
    complex c2(3,4);
    complex c3 = c1 + c2;
    c3.show();
    return 0;
}
```

- (●) 4 + i6
- ( ) 2 + i2
- ( ) Error
- ( ) Segmentation fault

Correct answer

- (●) Error

Given the following C++ code. How would you define the < operator for Box class so that when boxes b1 and b2 are compared in if block the program gives correct result?

```
#include <iostream>
#include <string>
using namespace std;
class Box
{
        int capacity;
     public:
        Box(){}
        Box(double capacity){
                this->capacity = capacity;
        }
};

int main(int argc, char const *argv[])
{
        Box b1(10);
        Box b2 = Box(14);
        if(b1 < b2){
                cout<<"Box 2 has large capacity.";
        }
        else{
                cout<<"Box 1 has large capacity.";
        }
        return 0;
}
```

```
bool operator<(Box b)
{
    return this->capacity < b.capacity ? true : false;
}
```

◉ Option 1

```
bool operator<(Box b)
{
    return this->capacity > b.capacity ? true : false;
}
```

○ Option 2

```
bool operator<(Box b)
{
    return  b1 > b2 ? true : false;
}
```

```
bool operator<(Box b)
{
    return this < b ? true : false;
}
```

Option 3

Option 4

Which is the correct statement about operator overloading?

- Only arithmetic operators can be overloaded
- Only non-arithmetic operators can be overloaded
- Precedence of operators are changed after overlaoding
- Associativity and precedence of operators does not change

Which is called ternary operator?

- ?:
- &&
- | | |
- ===