**C++ Programming Questions and Answers – Large Objects**

1. How to store the large objects in c++ if it extends its allocated memory?  
a) memory heap  
b) stack  
c) queue  
d) stack & queue

Answer: a  
Explanation: Memory heap will store the large objects in c++ if it extends its allocated memory.

2. When we are using heap operations what do we need to do to save the memory?  
a) rename the objects  
b) delete the objects after processing  
c) both rename & delete the objects  
d) add the objects

Answer: b  
Explanation: When you allocate memory from the heap, you must remember to clean up objects when you’re done! Failure to do so is called a memory leak.

3. Which container in c++ will take large objects?  
a) string  
b) class  
c) vector  
d) string & class

Answer: c  
Explanation: Because the vector is mainly used to store large objects for the game programming and other operations etc.

4. What will be the output of the following C++ code?

1. #include <iostream>
2. using namespace std;
3. class sample
4. {
5. public:
6. sample()
7. {
8. cout << "X::X()" << endl;
9. }
10. sample( sample const & )
11. {
12. cout << "X::X( X const & )" << endl;
13. }
14. sample& operator=( sample const & )
15. {
16. cout << "X::operator=(X const &)" << endl;
17. }
18. };
19. sample f()
20. {
21. sample tmp;
22. return tmp;
23. }
24. int main()
25. {
26. sample x = f();
27. return 0;
28. }

a) X::operator=(X const &)  
b) X::X( X const & )  
c) X::X()  
d) X::operator

Answer: c  
Explanation: As we are passing the object without any attributes it will return as X::X().  
Output:

$ g++ large.cpp

$ a.out

X::X()

5. How to stop your program from eating so much ram?  
a) Find a way to work with the data one at a time  
b) Declare it in program memory, instead of on the stack  
c) Use the hard drive, instead of RAM  
d) All of the mentioned

Answer: d  
Explanation: Some of the ways to stop the program by consuming more ram. They are  
i) Find a way to work with the data one at a time  
ii) Declare it in program memory, instead of on the stack  
iii) Use the hard drive, instead of RAM

6. Which option is best to eliminate the memory problem?  
a) use smart pointers  
b) use raw pointers  
c) use virtual destructor  
d) use smart pointers & virtual destructor

Answer: d  
Explanation: Virtual destructor means is that the object is destructed in reverse order in which it was constructed and the smart pointer will delete the object from memory when the object goes out of scope.

7. What is the size of the heap?  
a) 10MB  
b) 500MB  
c) 1GB  
d) Size of the heap memory is limited by the size of the RAM and the swap memory

Answer: d  
Explanation: Size of the heap memory is limited by the size of the RAM and the swap memory.

8. How to unlimit the size of the stack?  
a) setrlimit()  
b) unlimit()  
c) both setrlimit() & unlimit()  
d) setflimit()

Answer: a  
Explanation: setrlimit() is used to unlimit the size of the stack.

9. In Linux, how do the heaps and stacks are managed?  
a) ram  
b) secondary memory  
c) virtual memory  
d) static memory

Answer: c  
Explanation: In virtual memory, We can keep track of all the objects and access them much faster than any another.

10. Which is used to pass the large objects in c++?  
a) pass by value  
b) pass by reference  
c) both pass by value & reference  
d) pass by name

Answer: b  
Explanation: Because by using pass by reference we need to pass only address location, So it can save a lot of memory.

# C++ Programming Questions and Answers – Essential Operators

1. What are the essential operators in c++?  
a) +  
b) |  
c) <=  
d) All of the mentioned

Answer: d  
Explanation: Essential operators in c++ are +, |, <=.

2. In which direction does the assignment operation will take place?  
a) left to right  
b) right to left  
c) top to bottom  
d) bottom to top

Answer: b  
Explanation: In assignment operation, the flow of execution will be from right to left only.

3. Pick out the compound assignment statement.  
a) a = a – 5  
b) a = a / b  
c) a -= 5  
d) a = a + 5

Answer: c  
Explanation: When we want to modify the value of a variable by performing an operation on the value currently stored, We will use compound assignment statement. In this option, a -=5 is equal to a = a-5.

4. What will be the output of the following C++ code?

1. #include <iostream>
2. using namespace std;
3. int main ()
4. {
5. int a, b;
6. a = 10;
7. b = 4;
8. a = b;
9. b = 7;
10. cout << "a:";
11. cout << a;
12. cout << " b:";
13. cout << b;
14. return 0;
15. }

a) a:4 b:7  
b) a:10 b:4  
c) a:4 b:10  
d) a:4 b:6

Answer: a  
Explanation: In this program, we are reassigning the values of a and b because of this we got the output as a:4 b:7  
Output:

$ g++ ess.cpp

$ a.out

a:4 b:7

5. What will be the output of the following C++ code?

1. #include <iostream>
2. using namespace std;
3. int main ()
4. {
5. int a, b, c;
6. a = 2;
7. b = 7;
8. c = (a > b) ? a : b;
9. cout << c;
10. return 0;
11. }

a) 2  
b) 7  
c) 9  
d) 14

Answer: b  
Explanation: We are using the ternary operator to evaluate this expression. It will return first option, if first condition is true otherwise it will return second  
Output:

$ g++ ess1.cpp

$ a.out

7

6. What will be the output of the following C++ code?

1. #include <iostream>
2. using namespace std;
3. int main()
4. {
5. int a = 0;
6. int b = 10;
7. if ( a && b )
8. {
9. cout << "true"<< endl ;
10. }
11. else
12. {
13. cout << "false"<< endl ;
14. }
15. return 0;
16. }

a) true  
b) false  
c) error  
d) 10

Answer: b  
Explanation: && is called as Logical AND operator, if there is no zero in the operand means, it will be true otherwise false.  
Output:

$ g++ ess2.cpp

$ a.out

false

7. What is the associativity of add(+);?  
a) right to left  
b) left to right  
c) right to left & left to right  
d) top to bottom

Answer: b  
Explanation: left to right is the associativity of add(+);.

8. What is the name of | operator?  
a) sizeof  
b) or  
c) and  
d) modulus

Answer: b  
Explanation: | operator is used to find the ‘or’ of given values.

9. Which operator is having the highest precedence in c++?  
a) array subscript  
b) Scope resolution operator  
c) static\_cast  
d) dynamic\_cast

Answer: b  
Explanation: Scope resolution operator is having the highest precedence in c++.

10. What will be the output of the following C++ code?

1. #include <iostream>
2. using namespace std;
3. int main()
4. {
5. int a = 20;
6. int b = 10;
7. int c = 15;
8. int d = 5;
9. int e;
10. e = a + b \* c / d;
11. cout << e << endl ;
12. return 0;
13. }

a) 50  
b) 60  
c) 70  
d) 90

Answer: a  
Explanation: In this program, the value e is evaluated by precedence ad we got the output as 50.  
Output:

$ g++ ess4.cpp

$ a.out

50

**C++ Programming Questions and Answers – Subscripting**

1. subscript operator is used to access which elements?  
a) string  
b) char  
c) array  
d) float

Answer: c  
Explanation: To access any element of an array we use following syntax array[i], where i is called subscript representing the ith element of an array, whereas no such cases in char and strings.

2. How many arguments will the subscript operator will take for overloading?  
a) 1  
b) 2  
c) 0  
d) as many as possible

Answer: a  
Explanation: The subscript operator overload takes only one argument, but it can be of any type.

3. Pick out the correct statement.  
a) subscript operator has a higher precedence than the assignment operator  
b) subscript operator has a lower precedence than the assignment operator  
c) subscript operator is used with string elements  
d) subscript operator is used with char elements

Answer: a  
Explanation: Subscription operator has more precedence otherwise if that is not the case then the statement var = arr[i] will be meaningless and will have no effect.

4. What will be the output of the following C++ code?

1. #include <iostream>
2. using namespace std;
3. const int SIZE = 10;
4. class safe
5. {
6. private:
7. int arr[SIZE];
8. public:
9. safe()
10. {
11. register int i;
12. for (i = 0; i < SIZE; i++)
13. {
14. arr[i] = i;
15. }
16. }
17. int &operator[](int i)
18. {
19. if (i > SIZE)
20. {
21. cout << "Index out of bounds" <<endl;
22. return arr[0];
23. }
24. return arr[i];
25. }
26. };
27. int main()
28. {
29. safe A;
30. cout << A[5];
31. cout << A[12];
32. return 0;

33. }

a)

5Index out of bounds

0

b) 40  
c) 50  
d) 51

Answer: a  
Explanation: In this program, We are returning the elements in the specified array location and if it is out of bound means it will return the first element.  
Output:

$ g++ sub.cpp

$ a.out

5Index out of bounds

0

5. What will be the output of the following C++ code?

1. #include <iostream>
2. using namespace std;
3. class numbers
4. {
5. private:
6. int m\_nValues[10];
7. public:
8. int& operator[] (const int nValue);
9. };
10. int& numbers::operator[](const int nValue)
11. {
12. return m\_nValues[nValue];
13. }
14. int main()
15. {
16. numbers N;
17. N[5] = 4;
18. cout << N[5];
19. return 0;
20. }

a) 5  
b) 4  
c) 3  
d) 6

Answer: b  
Explanation: In this program, We are getting the values and returning it by overloading the subscript operator.  
Output:

$ g++ sub1.cpp

$ a.out

4

6. What will be the output of the following C++ code?

1. #include <iostream>
2. using namespace std;
3. const int limit = 4;
4. class safearray
5. {
6. private:
7. int arr[limit];
8. public:
9. int& operator [](int n)
10. {
11. if (n == limit - 1)
12. {
13. int temp;
14. for (int i = 0; i < limit; i++)
15. {
16. if (arr[n + 1] > arr[n])
17. {
18. temp = arr[n];
19. arr[n] = arr[n + 1];
20. arr[n + 1] = temp;
21. }
22. }
23. }
24. return arr[n];
25. }
26. };
27. int main()
28. {
29. safearray sa1;
30. for(int j = 0; j < limit; j++)
31. sa1[j] = j\*10;
32. for(int j = 0; j < limit; j++)
33. {
34. int temp = sa1[j];
35. cout << "Element " << j << " is " << temp;
36. }
37. return 0;
38. }

a) 0102030  
b) 1020300  
c) 3020100  
d) error

Answer: a  
Explanation: In this program, we are returning the array element by the multiple of 10.  
Output:

$ g++ sub2.cpp

$ a.out

0102030

7. What will be the output of the following C++ code?

1. #include <iostream>
2. using namespace std;
3. class A
4. {
5. public:
6. int x;
7. A(int n = 0) : x(n) {};
8. int& operator[](int n)
9. {
10. cout << "0" ;
11. return x;
12. }
13. int operator[](int n) const
14. {
15. cout << "1" ;
16. return x;
17. }
18. };
19. void foo(const A& a)
20. {
21. int z = a[2];
22. }
23. int main()
24. {
25. A a(7);
26. a[3] = 8;
27. int z = a[2];
28. foo(a);
29. return 0;
30. }

a) 110  
b) 111  
c) 011  
d) 001

Answer: d  
Explanation: In this program, we overloading the operator[] by using subscript operator.  
Output:

$ g++ sub3.cpp

$ a.out

001

8. What will be the output of the following C++ code?

1. #include <iostream>
2. using namespace std;
3. class sample
4. {
5. private:
6. int\* i;
7. int j;
8. public:
9. sample (int j);
10. ~sample ();
11. int& operator [] (int n);
12. };
13. int& sample::operator [] (int n)
14. {
15. return i[n];
16. }
17. sample::sample (int j)
18. {
19. i = new int [j];
20. j = j;
21. }
22. sample::~sample ()
23. {
24. delete [] i;
25. }
26. int main ()
27. {
28. sample m (5);
29. m [0] = 25;
30. m [1] = 20;
31. m [2] = 15;
32. m [3] = 10;
33. m [4] = 5;
34. for (int n = 0; n < 5; ++ n)
35. cout << m [n];
36. return 0;
37. }

a) 252015105  
b) 510152025  
c) 51015  
d) 51015210

Answer: a  
Explanation: In this program, we are printing the array in the reverse order by using subscript operator.  
Output:

$ g++ sub4.cpp

$ a.out

252015105

9. What do we need to do to pointer for overloading the subscript operator?  
a) reference pointer  
b) dereference pointer  
c) store it in heap  
d) memory locator

Answer: b  
Explanation: If you have a pointer to an object of some class type that overloads the subscript operator, you have to dereference that pointer in order to free the memory.

10. What do we need to use when we have multiple subscripts?  
a) operator()  
b) operator[]  
c) operator  
d) operator<>

Answer: a  
Explanation: The reason is that operator[] always takes exactly one parameter, but operator() can take any number of parameters.

**C++ Programming Questions and Answers – Function Call**

1. What is the use of function call operator?  
a) overloading the methods  
b) overloading the objects  
c) overloading the parameters  
d) overloading the string

Answer: b  
Explanation: Overloading the objects is the use of function call operator.

2. Pick out the correct statement.  
a) virtual functions does not give the ability to write a templated function  
b) virtual functions does not give the ability to rewrite a templated function  
c) virtual functions does give the ability to write a templated function  
d) virtual functions does not give the ability to rewrite a simple function

Answer: a  
Explanation: Virtual functions does not give the ability to write a templated function.

3. What will happen when the function call operator is overloaded?  
a) It will not modify the functions  
b) It will modify the functions  
c) It will modify the object  
d) It will modify the operator to be interpreted

Answer: d  
Explanation: It will modifies how the operator is to be interpreted when applied to objects of a given type.

4. What will be the output of the following C++ code?

1. #include <iostream>
2. using namespace std;
3. class Distance
4. {
5. private:
6. int feet;
7. int inches;
8. public:
9. Distance()
10. {
11. feet = 0;
12. inches = 0;
13. }
14. Distance(int f, int i)
15. {
16. feet = f;
17. inches = i;
18. }
19. Distance operator()(int a, int b, int c)
20. {
21. Distance D;
22. D.feet = a + c + 10;
23. D.inches = b + c + 100 ;
24. return D;
25. }
26. void displayDistance()
27. {
28. cout << feet << inches << endl;
29. }
30. };
31. int main()
32. {
33. Distance D1(11, 10), D2;
34. cout << "First Distance : ";
35. D1.displayDistance();
36. D2 = D1(10, 10, 10);
37. cout << "Second Distance :";
38. D2.displayDistance();
39. return 0;
40. }

a) First Distance : 1110

Second Distance :30120

b) First Distance : 110

Second Distance :3020

c) First Distance : 1115

Second Distance :30125

d) pre> First Distance : 100  
Second Distance :30120

Answer: a  
Explanation: We are calculating the foot and inches by using the function call operator.  
Output:

$ g++ call.cpp

$ a.out

First Distance : 1110

Second Distance :30120

5. What will be the output of the following C++ code?

1. #include <iostream>
2. using namespace std;
3. void duplicate (int& a, int& b, int& c)
4. {
5. a \*= 2;
6. b \*= 2;
7. c \*= 2;
8. }
9. int main ()
10. {
11. int x = 1, y = 3, z = 7;
12. duplicate (x, y, z);
13. cout << x << y << z;
14. return 0;
15. }

a) 1468  
b) 2812  
c) 2614  
d) 2713

Answer: c  
Explanation: We are passing the values by reference and modified the data on the function block.  
Output:

$ g++ call1.cpp

$ a.out

2614

6. What will be the output of the following C++ code?

1. #include <iostream>
2. using namespace std;
3. class three\_d
4. {
5. int x, y, z;
6. public:
7. three\_d() { x = y = z = 0; }
8. three\_d(int i, int j, int k) { x = i; y = j; z = k; }
9. three\_d operator()(three\_d obj);
10. three\_d operator()(int a, int b, int c);
11. friend ostream &operator<<(ostream &strm, three\_d op);
12. };
13. three\_d three\_d::operator()(three\_d obj)
14. {
15. three\_d temp;
16. temp.x = (x + obj.x) / 2;
17. temp.y = (y + obj.y) / 2;
18. temp.z = (z + obj.z) / 2;
19. return temp;
20. }
21. three\_d three\_d::operator()(int a, int b, int c)
22. {
23. three\_d temp;
24. temp.x = x + a;
25. temp.y = y + b;
26. temp.z = z + c;
27. return temp;
28. }
29. ostream &operator<<(ostream &strm, three\_d op) {
30. strm << op.x << ", " << op.y << ", " << op.z << endl;
31. return strm;
32. }
33. int main()
34. {
35. three\_d objA(1, 2, 3), objB(10, 10, 10), objC;
36. objC = objA(objB(100, 200, 300));
37. cout << objC;
38. return 0;
39. }

a) 55, 106, 156  
b) 55, 106  
c) 55, 106, 159  
d) 55, 106, 158

Answer: a  
Explanation: In this program, We are using the function call operator to calculate the value of objc.  
Output:

$ g++ call2.cpp

$ a.out

55, 106, 156

7. What will be the output of the following C++ code?

1. #include <iostream>
2. using namespace std;
3. class Complex
4. {
5. private:
6. float real;
7. float imag;
8. public:
9. Complex():real(0), imag(0){}
10. Complex operator ()(float re, float im)
11. {
12. real += re;
13. imag += im;
14. return \*this;
15. }
16. Complex operator() (float re)
17. {
18. real += re;
19. return \*this;
20. }
21. void display()
22. {
23. cout << "(" << real << "," << imag << ")" << endl;
24. }
25. };
26. int main()
27. {
28. Complex c1, c2;
29. c2 = c1(3.2, 5.3);
30. c1(6.5, 2.7);
31. c2(1.9);
32. cout << "c2=";c1.display();
33. cout << "c2=";c2.display();
34. return 0;
35. }

a) c2=(9.7,8)

c2=(5.1,5.3)

b) c2=(9,8)

c2=(5,5)

c)c2=(4.7,8)

c2=(2.1,5.3)

d) c2=(5.7,8)

c2=(5.1,5.6)

Answer: a  
Explanation: In this program, We are returning the real and imaginary part of the complex number by using function call operator.  
Output:

$ g++ call3.cpp

$ a.out

c2=(9.7,8)

c2=(5.1,5.3)

8. In which form does the function call operator can be overloaded?  
a) static member function  
b) non-static member function  
c) dynamis\_cast  
d) static\_cast

Answer: b  
Explanation: In non-static member function, the function call operator can be overloaded.

9. What will be the output of the following C++ code?

1. #include <iostream>
2. using namespace std;
3. int operate (int a, int b)
4. {
5. return (a \* b);
6. }
7. float operate (float a, float b)
8. {
9. return (a / b);
10. }
11. int main ()
12. {
13. int x = 5, y = 2;
14. float n = 5.0, m = 2.0;
15. cout << operate (x, y);
16. cout << operate (n, m);
17. return 0;
18. }

a) 119  
b) 102.5  
c) 123.4  
d) 128.4

Answer: b  
Explanation: In this program, We are overloading the function and getting the output as 10 and 2.5 by division and multiplication.  
Output:

$ g++ call3.cpp

$ a.out

102.5

10. What is the use of functor?  
a) It makes the object “callable” like a function  
b) It makes the class “callable” like a function  
c) It makes the attribute “callable” like a function  
d) It makes the argument “callable” like a function

Answer: a  
Explanation: functor makes the object “callable” like a function.

**C++ Programming Questions and Answers – Dereferencing**

1. Which is used to tell the computer that where a pointer is pointing to?  
a) dereference  
b) reference  
c) heap operations  
d) binary operations

Answer: a  
Explanation: dereference is used to tell the computer where a pointer is pointing to it.

2. Which is used to do the dereferencing?  
a) pointer without asterix  
b) value without asterix  
c) pointer with asterix  
d) value with asterix

Answer: c  
Explanation: Dereferencing is using a pointer with asterix. For example, \*(abc).

3. Pick out the correct option.  
a) References automatically dereference without needing an extra character  
b) References automatically dereference with an extra character  
c) Reference will not dereference  
d) Reference automatically dereference with extra space and character

Answer: a  
Explanation: References automatically dereference without needing an extra character.

4. What will be the output of the following C++ code?

1. #include <iostream>
2. using namespace std;
3. int main()
4. {
5. int a, b;
6. int\* c;
7. c = &a;
8. a = 200;
9. b = 200;
10. \*c = 100;
11. b = \*c;
12. cout << \*c << " " << b;
13. return 0;
14. }

a) 100 200  
b) 100 0  
c) 200 200  
d) 100 100

Answer: d  
Explanation: In this program, We are making the assignments and invoking the both b and c values as 100 by dereference operator.  
Output:

$ g++ def.cpp

$ a.out

100 100

5. What will be the output of the following C++ code?

1. #include <iostream>
2. using namespace std;
3. int main()
4. {
5. int x;
6. int \*p;
7. x = 5;
8. p = &x;
9. cout << \*p;
10. return 0;
11. }

a) 5  
b) 10  
c) memory address  
d) 15

Answer: a  
Explanation: In this program, we are copying the memory location of x into p and then printing the value in the address.  
Output:

$ g++ def1.cpp

$ a.out

5

6. What will be the output of the following C++ code?

1. #include <iostream>
2. using namespace std;
3. int main ()
4. {
5. int a;
6. int \* ptr\_b;
7. int \*\* ptr\_c;
8. a = 1;
9. ptr\_b = &a;
10. ptr\_c = &ptr\_b;
11. cout << a << "**\n**";
12. cout << \*ptr\_b << "**\n**";
13. cout << \*ptr\_c << "**\n**";
14. cout << \*\*ptr\_c << "**\n**";
15. return 0;
16. }

a) 1

1

0xbffc9924

1

b)1

1

1

0xbffc9924

c)1

0xbffc9924

1

&1

d) 0xbffc9924

Answer: a  
Explanation: In this program, We are printing the values and memory address  
by using the pointer and dereference operator.  
Output:

$ g++ def2.cpp

$ a.out

1

1

0xbffc9924

1

7. What will be the output of the following C++ code?

1. #include <iostream>
2. using namespace std;
3. int main()
4. {
5. int x = 9;
6. int\* p = &x;
7. cout << sizeof(p);
8. return 0;
9. }

a) 4  
b) 2  
c) Depends on compiler  
d) 8

Answer: c  
Explanation: The size of a data type mainly depends on compiler only.  
Output:

$ g++ def3.cpp

$ a.out

4

8. What will be the output of the following C++ code?

1. #include <iostream>
2. using namespace std;
3. int main()
4. {
5. double arr[] = {5.0, 6.0, 7.0, 8.0};
6. double \*p = (arr+2);
7. cout << \*p << endl;
8. cout << arr << endl;
9. cout << \*(arr+3) << endl;
10. cout << \*(arr) << endl;
11. cout << \*arr+9 << endl;
12. return 0;
13. }

a)7

0xbf99fc98

8

5

14

b)7

8

0xbf99fc98

5

14

c) 0xbf99fc98  
d) 14

Answer: a  
Explanation: In this program, We are printing the values that are pointed by pointer and also the dereference operator.  
Output:

$ g++ def5.cpp

$ a.out

7

0xbf99fc98

8

5

14

9. What does the dereference operator will return?  
a) rvalue equivalent to the value at the pointer address  
b) lvalue equivalent to the value at the pointer address  
c) it will return nothing  
d) it will return boolean values

Answer: b  
Explanation: It operates on a pointer variable, and returns an l-value equivalent to the value at the pointer address.

10. Pick out the correct statement.  
a) the null pointer dereference occurs where a pointer that is expected to be a valid address but instead is equal to null  
b) the null pointer dereference occurs where a pointer that is expected to be a valid address but instead is equal to the memory address  
c) rvalue equivalent to the value at the pointer address  
d) null pointer will not return anything

Answer: a  
Explanation: The null pointer dereference occurs where a pointer that is expected to be a valid address but instead is equal to null.

**C++ Programming Questions and Answers – Increment and Decrement**

1. Which operator works only with integer variables?  
a) increment  
b) decrement  
c) both increment & decrement  
d) binary operator

Answer: c  
Explanation: Because increment and decrement operator increases increasing and decreasing values of values and no such things define in strings so cannot be used with strings. Also they cannot be used with floats and doubles because there is no way to fix how much the value should be increased or decreased if increment or decrement operator is applied on such variables. That’s why both these operators only works with integer values.

2. How many types are there in increment/decrement operator?  
a) 1  
b) 2  
c) 3  
d) 4

Answer: b  
Explanation: There are two types of increment/decrement. They are postfix and prefix.

3. Pick out the correct statement.  
a) Increment operator ++ adds 1 to its operand  
b) Increment operator ++ adds 2 to its operand  
c) Decrement operator ++ subtracts 1 to its operand  
d) Decrement operator ++ subtracts 3 to its operand

Answer: a  
Explanation: Increment operator are used to increase the values of any integer variable by 1.

4. What will be the output of the following C++ code?

1. #include <stdio.h>
2. #include<iostream>
3. using namespace std;
4. int main()
5. {
6. int a = 21;
7. int c ;
8. c = a++;
9. cout << c;
10. return 0;
11. }

a) 21  
b) 22  
c) 23  
d) 20

Answer: a  
Explanation: value of ‘a’ will be stored in c and then only it will be incremented.  
Output:

$ g++ incre.cpp

$ a.out

21

5. What will be the output of the following C++ code?

1. #include <stdio.h>
2. #include<iostream>
3. using namespace std;
4. int main()
5. {
6. int x = 5, y = 5;
7. cout << ++x << --y << endl;
8. return 0;
9. }

a) 55  
b) 64  
c) 46  
d) 45

Answer: b  
Explanation: The values will be pre increment and pre decrement, So it will print as 64.  
Output:

$ g++ incre2.cpp

$ a.out

64

6. What will be the output of the following C++ code?

1. #include <stdio.h>
2. #include<iostream>
3. using namespace std;
4. int main()
5. {
6. int x = 5, y = 5, z;
7. x = ++x; y = --y;
8. z = x++ + y--;
9. cout << z;
10. return 0;
11. }

a) 10  
b) 11  
c) 9  
d) 12

Answer: a  
Explanation: In this program, the increment and decrement of evaluation of z will not be accounted because they are post.  
Output:

$ g++ incre3.cpp

$ a.out

10

7. What will be the output of the following C++ code?

1. #include <stdio.h>
2. #include<iostream>
3. using namespace std;
4. int main()
5. {
6. int x = 5, y = 5, z;
7. x = ++x; y = --y;
8. z = x + ++x;
9. cout << z;
10. return 0;
11. }

a) 11  
b) 12  
c) 13  
d) 14

Answer: d  
Explanation: In this program, we are adding the x value after pre incrementing two times.  
Output:

$ g++ incre4.cpp

$ a.out

14

8. What will be the output of the following C++ code?

1. #include <stdio.h>
2. #include<iostream>
3. using namespace std;
4. int main()
5. {
6. int num1 = 5;
7. int num2 = 3;
8. int num3 = 2;
9. num1 = num2++;
10. num2 = --num3;
11. cout << num1 << num2 << num3;
12. return 0;
13. }

a) 532  
b) 235  
c) 312  
d) 311

Answer: d  
Explanation: In this program, We are pre increment and post incrementing the operands and saving it.  
Output:

$ g++ incre5.cpp

$ a.out

311

9. Pick out the correct statement.  
a) Pre Increment is faster than post-increment  
b) post-increment is faster than Pre Increment  
c) pre increment is slower than post-increment  
d) pre decrement is slower than post-increment

Answer: a  
Explanation: Because Pre Increment take one-byte instruction & post increment takes two-byte instruction.

10. Which concepts does the Pre Increment use?  
a) call by value  
b) call by reference  
c) queue  
d) call by name

Answer: b  
Explanation: call by reference because the changes are reflected back to the same memory cells/variables.

**C++ Programming Questions and Answers – String Class**

1. How many types of representation are in the string?  
a) 1  
b) 2  
c) 3  
d) 4

Answer: b  
Explanation: C++ provides the following two types of string representations. They are C-style character string and string class type with Standard C++.

2. What is the header file for the string class?  
a) #include<ios>  
b) #include<str>  
c) #include<string>  
d) #include<stio>

Answer: c  
Explanation: #include<string> is the header file for the string class.

3. Which is used to return the number of characters in the string?  
a) length  
b) size  
c) both size & length  
d) name

Answer: c  
Explanation: Both will return the number of characters that conform to the string’s content.

4. What will be the output of the following C++ code?

1. #include <iostream>
2. #include <cstring>
3. using namespace std;
4. int main ()
5. {
6. char str1[10] = "Hello";
7. char str2[10] = "World";
8. char str3[10];
9. int len ;
10. strcpy( str3, str1);
11. strcat( str1, str2);
12. len = strlen(str1);
13. cout << len << endl;
14. return 0;
15. }

a) 5  
b) 55  
c) 11  
d) 10

Answer: d  
Explanation: In the program, We are concatenating the str1 and str2 and printing  
it’s total length. So the length is 10.  
Output:

$ g++ stri.cpp

$ a.out

10

5. What will be the output of the following C++ code?

1. #include <iostream>
2. #include <string>
3. using namespace std;
4. int main ()
5. {
6. string str ("microsoft");
7. string::reverse\_iterator r;
8. for (r = str.rbegin() ; r < str.rend(); r++ )
9. cout << \*r;
10. return 0;
11. }

a) microsoft  
b) micro  
c) tfosorcim  
d) tfos

Answer: c  
Explanation: ‘rbegin’ is used to reverse the given the string.  
Output:

$ g++ stri1.cpp

$ a.out

tfosorcim

6. What will be the output of the following C++ code?

1. #include <iostream>
2. #include <string>
3. using namespace std;
4. int main ()
5. {
6. string str ("nobody does like this");
7. string key ("nobody");
8. size\_t f;
9. f = str.rfind(key);
10. if (f != string::npos)
11. str.replace (f, key.length(), "everybody");
12. cout << str << endl;
13. return 0;
14. }

a) nobody does like this  
b) nobody  
c) everybody  
d) everybody does like this

Answer: d  
Explanation: rfind is used to find the characters in the string and replace is used to replace with certain characters.  
Output:

$ g++ stri2.cpp

$ a.out

everybody does like this

7. What will be the output of the following C++ code?

1. #include <iostream>
2. #include <string>
3. using namespace std;
4. int main ()
5. {
6. string str ("steve jobs is legend");
7. string::iterator it;
8. str.erase (str.begin()+ 5, str.end()-7);
9. cout << str << endl;
10. return 0;
11. }

a) jobs is  
b) steve legend  
c) steve  
d) steve jobs is

Answer: b  
Explanation: In this program, We are leaving the first 5 characters and last 7 characters and we are erasing the remaining the characters.  
Output:

$ g++ stri3.cpp

$ a.out

steve legend

8. What will be the output of the following C++ code?

1. #include <iostream>
2. #include <string>
3. using namespace std;
4. int main ()
5. {
6. string str ("Microsoft");
7. for (size\_t i = 0; i < str.length();)
8. {
9. cout << str.at(i-1);
10. }
11. return 0;
12. }

a) M  
b) Microsoft  
c) Micro  
d) runtime error

Answer: d  
Explanation: This program will terminate because the cout element is out of range.

9. What will be the output of the following C++ code?

1. #include <iostream>
2. #include <string>
3. using namespace std;
4. int main ()
5. {
6. string str ("Ubuntu");
7. cout << str.capacity();
8. cout << str.max\_size();
9. return 0;
10. }

a) 61073741820  
b) 51073741820  
c) 6 and max size depends on compiler  
d)15

9223372036854775807

Answer: d  
Explanation: str.capacity() returns the size of the storage space currently allocated for the string, expressed in terms of bytes and capacity of the string may be equal or greater. The max\_size() returns the max size of the string.  
Output:

$ g++ stri5.cpp

$ a.out

15

9223372036854775807

10. Which method do we use to append more than one character at a time?  
a) append  
b) operator+=  
c) data  
d) both append & operator+=

Answer: d  
Explanation: C++ allows to append more characters to string using both inbuilt append() function and using operator overloaded += operator.

**C++ Programming Questions and Answers – String – 1**

1. What is string objects in C++?  
a) Stream of alphabets  
b) A stream of well-defined characters  
c) Stream of characters  
d) A stream of characters terminated by \0

Answer: b  
Explanation: String is defined as streams of characters, not necessarily terminated by \0. Also, a string can contain characters other than alphabets.

2. What is Character-Array?  
a) array of alphabets  
b) array of well-defined characters  
c) array of characters  
d) array of characters terminated by \0

Answer: c  
Explanation: Character-Array is defined as an array of characters, not necessarily terminated by \0. Also, a character-array can contain characters other than alphabets.

3. Pick the incorrect statement about Character-Array.  
a) Character-Array can be terminated by a null character(‘\0’)  
b) Character-Array has a static size  
c) Character-Array has a dynamic size  
d) Character-Array has a threat of array-decay

Answer: c  
Explanation: As Character-Array is an array, its size should be defined during its declaration hence the size of Character-Array is static. A Character-Array is not necessarily to be terminated by a null character. Also, it has a threat of array-decay.

4. Pick the correct statement about string objects in C++.  
a) String objects must be terminated by a null character(‘\0’)  
b) String objects have a static size  
c) String objects have a dynamic size  
d) String objects use extra memory than required.

Answer: c  
Explanation: String objects are dynamic in nature i.e. their size varies as their value changes so they don’t use any extra memory and it is not necessary to terminate a string object by ‘\0’.

5. What will be the output of the following C++ code?

#include <iostream>

#include <string>

using namespace std;

int main(int argc, char const \*argv[])

{

string str;

cin>>str;

cout<<str;

return 0;

}

a) str  
b) Input provided by the user  
c) Error  
d) Segmentation fault

Answer: b  
Explanation: There is no error in the program and as we are asking the user to enter a string and printing that string to console. Therefore output will be the string provided by the user.

6. What will be the output of the following C++ code?

#include <iostream>

#include <string>

using namespace std;

int main(int argc, char const \*argv[])

{

char str[] = "Hello World";

cout<<str[0];

return 0;

}

a) H  
b) e  
c) Error  
d) o

Answer: a  
Explanation: The program has no errors so and as str = “Hello World” and we are trying to print the first character of str. Hence “H” is the answer.

7. What will be the output of the following C++ code?

#include <iostream>

#include <string>

using namespace std;

int main(int argc, char const \*argv[])

{

char str[10];

cin>>str;

cout<<str;

return 0;

}

a) Compiler-time Error  
b) Run-time Error  
c) Input given by the user  
d) Depends on the length of the string entered by the user

Answer: d  
Explanation: As the character array size is 10 so if the string entered by the user is <= 10 then there will be no error and the program runs perfectly otherwise if the length is > 10 then the program gives a run-time error because the string crosses the allocated memory space.  
Output:

length < 10

$ ./a.out

Hello

Hello

length > 10

$ ./a.out

C++Programming

\*\*\* stack smashing detected \*\*\*: terminated

Aborted (core dumped)

8. What will be the output of the following C++ code if the string entered by the user is “Hello World”?

#include <iostream>

#include <string>

using namespace std;

int main(int argc, char const \*argv[])

{

string str;

cin>>str;

cout<<str;

return 0;

}

a) Hello World  
b) Hello  
c) World  
d) Error

Answer: b  
Explanation: As cin considers \n or space as the terminating symbols for the input so when the user enters “Hello World” so only “Hello” will be stored into the str variable as cin stops scanning input after space.  
Output:

$ ./a.out

Hello World

Hello

9. Which header file is used to include the string object functions in C++?  
a) #include <string.h>  
b) #include <cstring>  
c) #include <string>  
d) #include <string.cpp>

Answer: c  
Explanation: #include <string> header file is used as it contains all the string object functions.

10. What will be the output of the following C++ code?

#include <iostream>

#include <string>

using namespace std;

int main(int argc, char const \*argv[])

{

char s1[6] = "Hello";

char s2[6] = "World";

char s3[12] = s1 + " " + s2;

cout<<s3;

return 0;

}

a) Hello World  
b) Hello  
c) World  
d) Error

Answer: d  
Explanation: There is no operation defined for the addition of character array in C++ hence the compiler throws an error as it does not understoods what to do about this expression.

11. What will be the output of the following C++ code?

#include <iostream>

#include <string>

using namespace std;

int main(int argc, char const \*argv[])

{

string s1 = "Hello";

string s2 = "World";

string s3 = s1 + " " + s2;

cout<<s3;

return 0;

}

a) Hello World  
b) Hello  
c) World  
d) Error

Answer: a  
Explanation: The program runs perfectly as string class has defined the addition of two strings so when two strings are added then both the strings are concatenated. Hence the output is “Hello World”.

12. Which of the following is correct way of concatenating two string objects in C++?

way 1:

string s1 = "hello";

string s2 = "world";

string s3 = s1 + s2;

way 2:

string s1 = "hello";

string s2 = "world";

string s3 = s1.append(s2);

way 3:

string s1 = "hello";

string s2 = "world";

string s3 = strcat(s1,s2);

a) 1 and 2  
b) 2 and 3  
c) 1 and 3  
d) 1, 2 and 3

Answer: a  
Explanation: To concatenate two string objects we are provided with either direct addition or append() function in string class but strcat() is char\* function hence they cannot be used to concatenate two string objects.

13. Which of the following is not a modifier function in string class?  
a) operator+=()  
b) operator[]()  
c) push\_back()  
d) erase()

Answer: b  
Explanation: [] operator is used to access one of the characters of the string objects whereas other functions are used to modify the string in some way.

14. Which function is used to get the length of a string object?  
a) str.length()  
b) str.size()  
c) str.max\_size()  
d) both size() and length() function

Answer: d  
Explanation: Both size() and length() are used to get the size of the string objects.

15. What is the identifier given to string class to declare string objects?  
a) String  
b) string  
c) STRING  
d) Any of the above can be used

Answer: b  
Explanation: string identifier is used as the name of the class string.

**C++ Programming Questions and Answers – String – 2**

1. What will be the output of the following C++ code?

#include <iostream>

#include <string>

#include <cstring>

using namespace std;

int main(int argc, char const \*argv[])

{

const char \*a = "Hello**\0**World";

cout<<a;

return 0;

}

a) Hello World  
b) Hello  
c) World  
d) Error

Answer: b  
Explanation: char\* are terminated by a ‘\0’ character so the string “Hello\0World” will be cut down to “Hello”.

2. What will be the output of the following C++ code?

#include <iostream>

#include <string>

#include <cstring>

using namespace std;

int main(int argc, char const \*argv[])

{

string s("a");

cout<<s;

return 0;

}

a) a  
b) empty string  
c) Error  
d) Segmentation fault

Answer: a  
Explanation: string class has a constructor for this call hence the string s will be assigned “a”.

3. What will be the output of the following C++ code?

#include <iostream>

#include <string>

#include <cstring>

using namespace std;

int main()

{

string s('a');

cout<<s;

return 0;

}

a) a  
b) empty string  
c) Error  
d) Segmentation fault

Answer: c  
Explanation: The string class provides string(string s) as a constructor not the string(char) as a constructor therefore this assignment is not valid.

4. Which is the correct way of concatenating a character at the end of a string object?

way 1:

string s;

s = s + 'a';

way 2:

string s;

s.push\_back('a');

a) 1 only  
b) 2 only  
c) both of them  
d) both are wrong

Answer: c  
Explanation: string class provides the addition of char and string and also push\_back(char) function to append a character at the end of a string.

5. What will be the output of the following C++ code?

#include <iostream>

#include <string>

using namespace std;

int main ()

{

std::string str ("Sanfoundry.");

str.back() = '!';

std::cout << str << endl;

return 0;

}

a) Sanfoundry.!  
b) Sanfoundry.  
c) Sanfoundry!  
d) Sanfoundry!.

Answer: c  
Explanation: back() function modifies the last character of the string with the character provided.

6. What will be the output of the following C++ code?

#include <iostream>

#include <string>

using namespace std;

int main ()

{

string str ("sanfoundry.");

str.front() = 'S';

cout << str << endl;

return 0;

}

a) Sanfoundry  
b) Sanfoundry.  
c) sanfoundry  
d) sanfoundry.

Answer: b  
Explanation: front() modifies the first character of the string with the character provided.

7. What will be the output of the following C++ code?

#include <iostream>

#include <string>

using namespace std;

int main ()

{

string str ("sanfoundry.");

cout << str.substr(3).substr(4) << endl;

return 0;

}

a) foundry.  
b) dry.  
c) oundry.  
d) found

Answer: b  
Explanation: As we are first taking the substring of s from 3 to end then on that substring we are taking substr from 4 to end which is equal to “dry.”.

8. What will be the output of the following C++ code?

#include <iostream>

#include <string>

using namespace std;

int main ()

{

string str = "Sanfoundry!";

cout<<str.capacity();

cout<<str.size();

return 0;

}

a) 1511  
b) 1111  
c) 1115  
d) 010

Answer: a  
Explanation: Capacity of a string object is defined as the length of string plus the extra space given to that object which will be used further if string is expanded.

**C++ Programming MCQ – Constructors and Destructors**

1. What is the role of a constructor in classes?  
a) To modify the data whenever required  
b) To destroy an object  
c) To initialize the data members of an object when it is created  
d) To call private functions from the outer world

Answer: c  
Explanation: A constructor is used in classes to initialize data members of class in order to avoid errors/segmentation faults.

2. Why constructors are efficient instead of a function init() defined by the user to initialize the data members of an object?  
a) Because user may forget to call init() using that object leading segmentation fault  
b) Because user may call init() more than once which leads to overwriting values  
c) Because user may forget to define init() function  
d) All of the mentioned

Answer: d  
Explanation: We cannot use init() because as mentioned in options that user may forget to initialize the members which will lead to a segmentation fault. Also if the user calls the init() function more than once it may overwrite the values and may result into disastrous results. Also if any user forgets to define init() function then no object will be initialized whereas if any constructor is not defined in any class the class provides a default constructor for initialization.

3. What is a copy constructor?  
a) A constructor that allows a user to move data from one object to another  
b) A constructor to initialize an object with the values of another object  
c) A constructor to check the whether to objects are equal or not  
d) A constructor to kill other copies of a given object.

Answer: b  
Explanation: Copy constructor allows the user to initialize an object with the values of another object instead of supplying the same set of values again to initialize the object.

4. What will be the output of the following C++ code?

#include <iostream>

#include <string>

using namespace std;

class A{

int a;

public:

A(int i){

a = i;

}

void assign(int i){

a = i;

}

int return\_value(){

return a;

}

};

int main(int argc, char const \*argv[])

{

A obj;

obj.assign(5);

cout<<obj.return\_value();

}

a) 5  
b) 55  
c) Error  
d) Segmentation Fault

Answer: c  
Explanation: As we have defined a constructor which takes an int parameter, so when we are trying to declare an object obj of class A without supplying any parameter then as a constructor is overwritten it will give an error saying that no matching function found. So whenever one writes a constructor then the default constructor is overwritten hence if you want to declare an object without parameter then you also have to define that constructor.

5. What will be the output of the following C++ code?

#include <iostream>

#include <string>

using namespace std;

class A{

int a;

A(){

a = 5;

}

public:

void assign(int i){

a = i;

}

int return\_value(){

return a;

}

};

int main(int argc, char const \*argv[])

{

A obj;

obj.assign(10);

cout<<obj.return\_value();

}

a) 5  
b) 10  
c) Error  
d) Segmentation fault

Answer: c  
Explanation: Here the constructor is made private and as no object can access any private object directly therefore the program will give error. One should always define a constructor as public.

6. In the following C++ code how many times the string “A’s constructor called” will be printed?

#include <iostream>

#include <string>

using namespace std;

class A{

int a;

public:

A(){

cout<<"A's constructor called";

}

};

class B{

static A a;

public:

B(){

cout<<"B's constructor called";

}

static A get(){

return a;

}

};

A B::a;

int main(int argc, char const \*argv[])

{

B b;

A a1 = b.get();

A a2 = b.get();

A a3 = b.get();

}

a) 3  
b) 4  
c) 2  
d) 1

Answer: d  
Explanation: As the object is defined ony once in the program at line A B::a, so the constructor of A is called only once. For objects a1, a2 and a3 copy constructor is called so the string will not be printed for them.

7. What happens if a user forgets to define a constructor inside a class?  
a) Error occurs  
b) Segmentation fault  
c) Objects are not created properly  
d) Compiler provides a default constructor to avoid faults/errors

Answer: d  
Explanation: The C++ compiler always provides a default constructor if one forgets to define a constructor inside a class.

8. How many parameters does a default constructor require?  
a) 1  
b) 2  
c) 0  
d) 3

Answer: c  
Explanation: A default constructor does not require any parameters for object creation that’s why sometimes we declare an object without any parameters.

9. How constructors are different from other member functions of the class?  
a) Constructor has the same name as the class itself  
b) Constructors do not return anything  
c) Constructors are automatically called when an object is created  
d) All of the mentioned

Answer: d  
Explanation: All the above mention are the reasons where constructor differs from other normal member functions of a class.

10. How many types of constructors are there in C++?  
a) 1  
b) 2  
c) 3  
d) 4

Answer: c  
Explanation: There are three types of constructors in C++ namely default, parameterized and copy constructor.

11. What will be the output of the following C++ code?

#include <iostream>

#include <string>

using namespace std;

class A{

mutable int a;

public:

A(){

cout<<"Default constructor called**\n**";

}

A(const A& a){

cout<<"Copy Constructor called**\n**";

}

};

int main(int argc, char const \*argv[])

{

A obj;

A a1 = obj;

A a2(obj);

}

a)Default constructor called

Copy Constructor called

b)Default constructor called

Copy Constructor called

Copy Constructor called

c)Default constructor called

Default constructor called

Copy Constructor called

d)Copy Constructor called

Default constructor called

Copy Constructor called

Answer: b  
Explanation: When object obj is declared then the default constructor is called. When we are declaring the a1 object as we are equating obj to a1 object obj will be copied to a1 hence copy constructor is called, similarly when object a2 is created obj is passed as a parameter to function which is available as copy constructor function, hence copy constructor will be called. So one time Default constructor and two times copy constructor.

12. What will be the output of the following C++ code?

#include <iostream>

#include <string>

using namespace std;

class A{

mutable int a;

public:

A(){

cout<<"A's default constructor called**\n**";

}

A(const A& a){

cout<<"A's copy Constructor called**\n**";

}

};

class B{

A obj;

public:

B(){

cout<<"B's Constructor called**\n**";

}

};

int main(int argc, char const \*argv[])

{

B b1;

B b2;

}

a)B's Constructor called

B's Constructor called

b)B's Constructor called

A's default constructor called

B's Constructor called

A's default constructor called

c)A's default constructor called

B's Constructor called

A's default constructor called

B's Constructor called

d)A's default constructor called

B's Constructor called

A's copy Constructor called

Answer: c  
Explanation: Here when we are declaring the object b1 of class B then first the constructor of class B will be called, in which first it will initialize all the members of class B and as obj from class A is member of class B and it should be initialized so the A’s default constructor will be called and terminates after that B’s constructor terminates hence A’s default constructor called is printed before B’s constructor called.

13. What is the role of destructors in Classes?  
a) To modify the data whenever required  
b) To destroy an object when the lifetime of an object ends  
c) To initialize the data members of an object when it is created  
d) To call private functions from the outer world

Answer: b  
Explanation: Destructors are used in Classes to destroy an object after its lifetime is over i.e. to free resources occupied by that object.

14. What is syntax of defining a destructor of class A?  
a) A(){}  
b) ~A(){}  
c) A::A(){}  
d) ~A(){};

Answer: b  
Explanation: A destructor starts with a ~(tilde) symbol, has the same name as the class.

15. When destructors are called?  
a) When a program ends  
b) When a function ends  
c) When a delete operator is used  
d) All of the mentioned

Answer: d  
Explanation: Destructors are called at the following time:  
i) at the end of the program to destroy objects declared in the main() or global scope.  
ii) at the end of a function to destroy objects declared at that function scope.  
iii) when user by himself tries to delete an object using the delete operator.  
iv) at the end of a block to destroy objects declared at that block scope.

**C++ Programming MCQ – Constructors and Destructors – 2**

1. What is the difference between constructors and destructors?  
a) They have a different function name  
b) Constructors does not have return type whereas destructors do have  
c) Constructors allow function parameters whereas destructors do not  
d) Constructors does not function parameters

Answer: c  
Explanation: Both the constructors and destructors have the same function name and both of them do not have return type but constructors allow function parameters whereas destructors do not.

2. How many Destructors are allowed in a Class?  
a) 1  
b) 2  
c) 3

d) Any number

Answer: a  
Explanation: A class in C++ allows only one destructor, which is called whenever the lifetime of an object ends.

3. What will be the output of the following C++ code?

#include <iostream>

#include <string>

using namespace std;

class A{

mutable int a;

public:

A(){

cout<<"A's Constructor called**\n**";

}

~A(){

cout<<"A's Destructor called**\n**";

}

};

class B{

A a;

public:

B(){

cout<<"B's Constructor called**\n**";

}

~B(){

cout<<"B's Destructor called**\n**";

}

};

int main(int argc, char const \*argv[])

{

B b1;

}

a)A's Constructor called

B's Constructor called

b)A's Destructor called

B's Destructor called

c)A's Constructor called

B's Constructor called

B's Destructor called

A's Destructor called

d)A's Constructor called

B's Constructor called

A's Destructor called

B's Destructor called

Answer: c  
Explanation: The destructors for an object is called before the destructor of its data members or bases.

4. What will be the output of the following C++ code?

#include <iostream>

#include <string>

using namespace std;

class A{

mutable int a;

public:

A(){

cout<<"A's Constructor called**\n**";

}

~A(){

cout<<"A's Destructor called**\n**";

}

};

class B: public A{

public:

B(){

cout<<"B's Constructor called**\n**";

}

~B(){

cout<<"B's Destructor called**\n**";

}

};

int main(int argc, char const \*argv[])

{

B b1;

}

a)A's Constructor called

B's Constructor called

b)A's Destructor called

B's Destructor called

c)A's Constructor called

B's Constructor called

B's Destructor called

A's Destructor called

d)A's Constructor called

B's Constructor called

A's Destructor called

B's Destructor called

Answer: c  
Explanation: Though B class have no data member of the class but as class B is derived from class A, the destructor of class A will be called to destroy the data inherited from class A to class B.

 5. What will be the output of the following C++ code?

#include <iostream>

#include <string>

using namespace std;

class A{

mutable int a;

public:

A(){

cout<<"A's Constructor called**\n**";

}

~A(){

cout<<"A's Destructor called**\n**";

}

};

class B: public A{

A a;

public:

B(){

cout<<"B's Constructor called**\n**";

}

~B(){

cout<<"B's Destructor called**\n**";

}

};

int main(int argc, char const \*argv[])

{

B b1;

}

a)A's Constructor called

B's Constructor called

b)A's Destructor called

B's Destructor called

c)A's Constructor called

B's Constructor called

B's Destructor called

A's Destructor called

d)A's Constructor called

B's Constructor called

A's Destructor called

B's Destructor called

Answer: c  
Explanation: There are two calls to constructor of class A, one is for the data member of class B and second because class B is derived from class A. Similarly two destructor calls.

6. What will be the output of the following C++ code?

#include <iostream>

#include <string>

using namespace std;

class A{

mutable int a;

public:

A(){

cout<<"A's Constructor called**\n**";

}

~A(){

cout<<"A's Destructor called**\n**";

}

};

class B{

static A a;

public:

B(){

cout<<"B's Constructor called**\n**";

}

~B(){

cout<<"B's Destructor called**\n**";

}

};

int main(int argc, char const \*argv[])

{

B b1;

}

a)A's Constructor called

B's Constructor called

b)B's Constructor called

B's Destructor called

c)A's Constructor called

B's Constructor called

B's Destructor called

A's Destructor called

d)A's Constructor called

B's Constructor called

A's Destructor called

B's Destructor called

Answer: b  
Explanation: Here as ‘a’ is a static member of class B and as all static members should be initialized separately as no object creation initializes static member and as ‘a’ is not initialized, hence no call will be made to the constructor of class A.

7. What will be the output of the following C++ code?

#include <iostream>

#include <string>

using namespace std;

class A{

mutable int a;

public:

A(){

cout<<"A's Constructor called**\n**";

}

~A(){

cout<<"A's Destructor called**\n**";

}

};

class B{

static A a;

public:

B(){

cout<<"B's Constructor called**\n**";

}

~B(){

cout<<"B's Destructor called**\n**";

}

};

A B::a;

int main(int argc, char const \*argv[])

{

return 0;

}

a)A's Constructor called

A's Destructor called

b)B's Constructor called

B's Destructor called

c)A's Constructor called

B's Constructor called

B's Destructor called

A's Destructor called

d)A's Constructor called

B's Constructor called

A's Destructor called

B's Destructor called

Answer: a  
Explanation: Here as no object of B is declared so no call to B’s constructor but as we have initialised the static member ‘a’ of class B, hence A’s constructor and destructor will be called once.

8. Which of the following represents the correct explicit call to a constructor of class A?

class A{

int a;

public:

A(int i)

{

a = i;

}

}

1. A a(5);
2. A a;
3. A a = A(5);
4. A a = A();

Answer: c  
Explanation: Explicit call represents the programmer by himself mentioning the type name. So A a = A(5); is the correct explicit call as we are mentioning typename A(5) from our side, whereas A a = A(); is not the correct call because no such constructor is there in class A.

**C++ Programming MCQ – Constructors and Destructors – 3**

1. Which of the following constructors are provided by the C++ compiler if not defined in a class?  
a) Default constructor  
b) Copy Assignment Operator  
c) Copy constructor  
d) All of the mentioned

Answer: d  
Explanation: If a programmer does not define the above constructors in a class the C++ compiler by default provides these constructors to avoid error on basic operations. The compiler will do a shallow copy for “Copy constructor” and “Copy Assignment Operator” if there are no user-defined “copy constructors” or “copy assignment operator” constructors.

2. When a copy constructor is called?  
a) When an object of the class is returned by value  
b) When an object of the class is passed by value to a function  
c) When an object is constructed based on another object of the same class  
d) All of the mentioned

Answer: d  
Explanation: Copy constructor is called in all the above-mentioned criteria because in all the above cases we are somehow trying to copy one object into another.

3. What will be the output of the following C++ code?

#include <iostream>

using namespace std;

class A{

A(){

cout<<"Constructor called";

}

};

int main(int argc, char const \*argv[])

{

A a;

return 0;

}

a) Constructor called  
b) Nothing printed  
c) Error  
d) Segmentation fault

Answer: c  
Explanation: No constructor should be made private because objects need to call them and as by default all the members of a class are private therefore constructor defined in the above program is private which is wrong therefore the compiler gives the error.

4. What will be the output of the following C++ code?

#include <iostream>

using namespace std;

class A{

public:

A(){

cout<<"Constructor called";

}

};

int main(int argc, char const \*argv[])

{

A \*a;

return 0;

}

a) Constructor called  
b) Nothing printed  
c) Error  
d) Segmentation fault

Answer: b  
Explanation: As we have declared a pointer variable for class A but we have not initialized the memory to that pointer and until the memory is not initialized the constructor for the pointer variable will not be called hence nothing is printed on the screen.

5. What will be the output of the following C++ code?

#include <iostream>

using namespace std;

class A{

public:

int a;

};

int main(int argc, char const \*argv[])

{

A a1 = {10};

A a2 = a1;

cout<<a1.a<<a2.a;

return 0;

}

a) 1010  
b) 87368746  
c) Error  
d) Segmentation fault

Answer: a  
Explanation: Although the declaration of object a1 looks erroneous but actually it is acceptable by the C++ compiler to take values for class objects as mentioned above. Next value of a1 is copied to a2 hence 1010 is printed.  
Output:  
$ ./a.out  
1010

6. What will be the output of the following C++ code?

#include <iostream>

using namespace std;

class A{

public:

int a;

A(int a){

this->a = a;

}

};

int main(int argc, char const \*argv[])

{

A a1, a2(10);

cout<<a2.a;

return 0;

}

a) 10  
b) Compile time error  
c) Run-time error  
d) No output

Answer: b  
Explanation: The declaration of object a1 needs a constructor without any arguments which is not available in the class as we have overwritten the default constructor, therefore, the program gives the error.

7. What will be the output of the following C++ code?

#include <iostream>

using namespace std;

class A{

public:

int a;

A(int a=0){

this->a = a;

}

};

int main(int argc, char const \*argv[])

{

A a1, a2(10);

cout<<a1.a<<a2.a;

return 0;

}

a) 010  
b) 100  
c) 001  
d) Error

Answer: a  
Explanation: As constructor is accepting default parameter therefore the declaration of a1 and a2 both are valid hence the program runs successfully.

8. What will be the output of the following C++ code?

#include <iostream>

using namespace std;

class A{

public:

int a;

A(){

cout<<"Constructor called";

}

};

int main(int argc, char const \*argv[])

{

A \*a1 = (A\*)malloc(sizeof(A));

return 0;

}

1. Constructor called
2. Nothing printed
3. Error
4. Segmentation fault

Answer: b  
Explanation: Unlike new malloc never calls the constructor of a class hence when we are assigning memory to an object of class A using malloc constructor is not called hence nothing is printed.

9. What will be the output of the following C++ code?

#include <iostream>

using namespace std;

class A{

public:

int a;

A(){

cout<<"Constructor called";

}

} a;

int main(int argc, char const \*argv[])

{

return 0;

}

a) Constructor called  
b) Nothing printed  
c) Error  
d) Segmentation fault

Answer: a  
Explanation: In this program, we have defined a global variable an outside main function for which constructor will be called hence the output is printed.

10. What will be the output of the following C++ code?

#include <iostream>

using namespace std;

class A{

public:

int a;

A(){

cout<<"Constructor called**\n**";

}

} a;

int main(int argc, char const \*argv[])

{

cout<<"Main function**\n**";

return 0;

}

a)Constructor called

Main function

b)Main function

Constructor called

c) Error  
d) Segmentation fault

Answer: a  
Explanation: In this program, we have defined a global variable an outside main function for which constructor will be called. Now as a is a global variable, therefore, the call for constructor will be before the main function hence constructor called will be printed before the Main function.

11. What will be the output of the following C++ code?

#include <iostream>

using namespace std;

class A{

A(){

cout<<"A's Constructor called**\n**";

}

};

class B

{

public:

A a;

B(){

cout<<"B's constructor called**\n**s";

}

};

int main(int argc, char const \*argv[])

{

B b;

return 0;

}

a)A's Constructor called

B's constructor called

b)B's Constructor called

A's constructor called

c) Error  
d) Segmentation fault

Answer: c  
Explanation: As constructor of class A is defined private and we are trying to define an object of class A which cannot call this constructor as it is private therefore the program gives an error.

12. What will be the output of the following C++ code?

#include <iostream>

using namespace std;

class A{

A(){

cout<<"A's Constructor called**\n**";

}

friend class B;

};

class B

{

public:

A a;

B(){

cout<<"B's constructor called**\n**s";

}

};

int main(int argc, char const \*argv[])

{

B b;

return 0;

}

a)A's Constructor called

B's constructor called

b)B's Constructor called

A's constructor called

c) Error  
d) Segmentation fault

Answer: a  
Explanation: Now still the constructor of a class is private but class B is friend class of A hence it can access the private members of class A and as in the above program we defining an object of class A in class B only, therefore, the program runs fine.

13. What will be the output of the following C++ code?

#include <iostream>

using namespace std;

class A{

~A(){}

};

class B

{

public:

A a;

};

int main(int argc, char const \*argv[])

{

B b;

return 0;

}

a)A's Constructor called

B's constructor called

b)B's Constructor called

A's constructor called

c) Error  
d) Segmentation fault

Answer: c  
Explanation: In this program, the destructor of class A is private therefore the destructor for object a cannot be called hence the program gives an error.

14. How destructor overloading is done?  
a) By changing the number of parameters  
b) By changing the parameters type  
c) By changing both the number of parameters and their type  
d) No chance for destructor overloading

Answer: d  
Explanation: A class is allowed to have only one destructor. Therefore there is no point of destructor overloading.

15. Which of the following is correct?  
a) Destructors can be virtual  
b) There can be more than one destructor in a class  
c) Destructor definition starts with !  
d) Destructor is used to initialize objects

Answer: a  
Explanation: Destructors can be virtual. They are used to destroy objects. Only one destructor is allowed per class. Destructor definition starts with a tilde(~).