**Quiz 02**

1. **What is the difference between struct and class in C++?**

|  |  |
| --- | --- |
| **A** | All members of a structure are public and structures don't have constructors and destructors |
| **B** | Members of a class are private by default and members of struct are public by default. When deriving a struct from a class/struct, default access-specifier for a base class/struct is public and when deriving a class, default access specifier is private |
| **C** | All members of a structure are public and structures don't have virtual functions |
| **D** | All of the above |

1. **Predict the output of following C++ program**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| *#include<iostream>*  *using namespace std;*    *class Empty {};*    *int main()*  *{*  *cout << sizeof(Empty);*  *return 0;*  *}*   |  |  | | --- | --- | | **A** | A non-zero value | | **B** | 0 | | **C** | Compiler Error | | **D** | Runtime Error | |
|  |

1. **Predict the output of following C++ program**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| *class Test {*  *int x;*  *};*  *int main()*  *{*  *Test t;*  *cout << t.x;*  *return* 0;  }   |  |  | | --- | --- | | **A** | 0 | | **B** | Garbage Value | | **C** | Compiler Error | |  |  | |

1. **Which of the following is true?**

|  |  |
| --- | --- |
| **A** | All objects of a class share all data members of class |
| **B** | Objects of a class do not share non-static members. Every object has its own copy. |
| **C** | Objects of a class do not share codes of non-static methods, they have their own copy |
| **D** | None of the above |

1. **Assume that an integer and a pointer each takes 4 bytes. Also, assume that there is no alignment in objects. Predict the output following program.**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| *#include<iostream>*  *using namespace std;*    *class Test*  *{*  *static int x;*  *int \*ptr;*  *int y;*  *};*    *int main()*  *{*  *Test t;*  *cout << sizeof(t) << " ";*  *cout << sizeof(Test \*);*  *}*   |  |  | | --- | --- | | **A** | 12 4 | | **B** | 12 12 | | **C** | 8 4 | | **D** | 8 8 | |

1. **Which of the following is true about the following program**

|  |
| --- |
| *#include <iostream>*  *class Test*  *{*  *public:*  *int i;*  *void get();*  *};*  *void Test::get()*  *{*  *std::cout << "Enter the value of i: ";*  *std::cin >> i;*  *}*  *Test t;  // Global object*  *int main()*  *{*  *Test t;  // local object*  *t.get();*  *std::cout << "value of i in local t: "<<t.i<<'\n';*  *::t.get();*  *std::cout << "value of i in global t: "<<::t.i<<'\n';*  *return 0;*  *}* |

|  |  |
| --- | --- |
| **A** | Compiler Error: Cannot have two objects with same class name |
| **B** | Compiler Error in Line "::t.get();" |
| **C** | Compiles and runs fine |

1. **Predict the output of following C++ program**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| *#include<iostream>*  *using namespace std;*  *class Point {*  *public:*  *Point() { cout << "Constructor called"; }*  *};*    *int main()*  *{*  *Point t1, \*t2;*  *return 0;*  *}*   |  |  | | --- | --- | | **A** | Compiler Error | | **B** | Constructor called Constructor called | | **C** | Constructor called | |  |  | |

1. **Predict the output of following C++ program**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  | | --- | | *#include<iostream>*  *using namespace std;*    *class Point {*  *public:*  *Point() { cout << "Normal Constructor called\n"; }*  *Point(const Point &t) { cout << "Copy constructor called\n"; }*  *};*    *int main()*  *{*  *Point \*t1, \*t2;*  *t1 = new Point();*  *t2 = new Point(\*t1);*  *Point t3 = \*t1;*  *Point t4;*  *t4 = t3;*  *return 0;*  *}* |  |  |  | | --- | --- | | **A** | Normal Constructor called Normal Constructor called Normal Constructor called Copy Constructor called Copy Constructor called Normal Constructor called Copy Constructor called | | **B** | Normal Constructor called Copy Constructor called Copy Constructor called  Normal Constructor called Copy Constructor called | | **C** | Normal Constructor called Copy Constructor called Copy Constructor called Normal Constructor called | |  |  | |

1. **Which of the following is true about the following program**

|  |
| --- |
| *#include <iostream>*  *using namespace std;*    *class X*  *{*  *public:*  *int x;*  *};*    *int main()*  *{*  *X a = {10};*  *X b = a;*  *cout << a.x << " " << b.x;*  *return 0;*  *}* |

|  |  |
| --- | --- |
| **A** | Compiler Error |
| **B** | 10 followed by Garbage Value |
| **C** | 10 10 |
| **D** | 10 0 |

1. **Which of the following is true about the following program**

|  |
| --- |
| *#include <iostream>*  *#include<iostream>*  *#include<string.h>*  *using namespace std;*    *class String*  *{*  *char \*str;*  *public:*  *String(const char \*s);*  *void change(int index, char c) { str[index] = c; }*  *char \*get() { return str; }*  *};*    *String::String(const char \*s)*  *{*  *int l = strlen(s);*  *str = new char[l+1];*  *strcpy(str, s);*  *}*    *int main()*  *{*  *String s1("geeksQuiz");*  *String s2 = s1;*  *s1.change(0, 'G');*  *cout << s1.get() << " ";*  *cout << s2.get();*  *}* |

|  |  |
| --- | --- |
| **A** | GeeksQuiz geeksQuiz |
| **B** | GeeksQuiz GeeksQuiz |
| **C** | geeksQuiz geeksQuiz |
| **D** | geeksQuiz GeeksQuiz |

1. **Which of the following is true about constructors.**

1) They cannot be virtual.

2) They cannot be private.

3) They are automatically called by new operator

|  |  |
| --- | --- |
| **A** | All 1, 2, and 3 |
| **B** | Only 1 and 3 |
| **C** | Only 1 and 2 |
| **D** | Only 2 and 3 |

1. **Predict the output of following C++ program**

|  |
| --- |
| *#include <iostream>*  *using namespace std;*    *int i;*    *class A*  *{*  *public:*  *~A()*  *{*  *i=10;*  *}*  *};*    *int foo()*  *{*  *i=3;*  *A ob;*  *return i;*  *}*    *int main()*  *{*  *cout << foo() << endl;*  *return 0;*  *}* |

|  |  |
| --- | --- |
| **A** | 0 |
| **B** | 3 |
| **C** | 10 |
| **D** | None of the above |

1. **Can destructors be virtual in C++?**

|  |
| --- |
|  |

|  |  |
| --- | --- |
| **A** | Yes |
| **B** | No |

1. **Predict the output of following C++ program**

|  |
| --- |
| *#include <iostream>*  *using namespace std;*  *class A*  *{*  *int id;*  *static int count;*  *public:*  *A() {*  *count++;*  *id = count;*  *cout << "constructor for id " << id << endl;*  *}*  *~A() {*  *cout << "destructor for id " << id << endl;*  *}*  *};*    *int A::count = 0;*    *int main() {*  *A a[3];*  *return 0;*  } |

|  |  |
| --- | --- |
| **A** | constructor for id 1  constructor for id 2  constructor for id 3  destructor for id 3  destructor for id 2  destructor for id 1 |
| **B** | constructor for id 1  constructor for id 2  constructor for id 3  destructor for id 1  destructor for id 2  destructor for id 3 |
| **C** | Compiler Dependent. |
| **D** | constructor for id 1  destructor for id 1 |

1. **Predict the output of following C++ program**

|  |
| --- |
| *#include <iostream>*  *using namespace std;*  *class A*  *{*  *protected:*  *int x;*  *public:*  *A() {x = 0;}*  *friend void show();*  *};*    *class B: public A*  *{*  *public:*  *B() : y (0) {}*  *private:*  *int y;*  *};*    *void show()*  *{*  *A a;*  *B b;*  *cout << "The default value of A::x = " << a.x << " ";*  *cout << "The default value of B::y = " << b.y;*  *}* |

|  |  |
| --- | --- |
| **A** | Compiler Error in show() because x is protected in class A |
| **B** | Compiler Error in show() because y is private in class b |
| **C** | The default value of A::x = 0 The default value of B::y = 0 |
| **D** | Compiler Dependent |

1. **Predict the output of following C++ program**

|  |
| --- |
| *#include <iostream>*  *using namespace std;*    *class B;*  *class A {*  *int a;*  *public:*  *A():a(0) { }*  *void show(A& x, B& y);*  *};*    *class B {*  *private:*  *int b;*  *public:*  *B():b(0) { }*  *friend void A::show(A& x, B& y);*  *};*    *void A::show(A& x, B& y) {*  *x.a = 10;*  *cout << "A::a=" << x.a << " B::b=" << y.b;*  *}*    *int main() {*  *A a;*  *B b;*  *a.show(a,b);*  *return 0;*  *}* |

|  |  |
| --- | --- |
| **A** | Compiler Error |
| **B** | A::a=10 B::b=0 |
| **C** | A::a=0 B::b=0 |
|  |  |

1. **Predict the output of following C++ program**

|  |
| --- |
| *#include <iostream>*  *using namespace std;*    *class Player*  *{*  *private:*  *int id;*  *static int next\_id;*  *public:*  *int getID() { return id; }*  *Player()  {  id = next\_id++; }*  *};*  *int Player::next\_id = 1;*    *int main()*  *{*  *Player p1;*  *Player p2;*  *Player p3;*  *cout << p1.getID() << " ";*  *cout << p2.getID() << " ";*  *cout << p3.getID();*  *return 0;*  *}* |

|  |  |
| --- | --- |
| **A** | Compiler Error |
| **B** | 1 2 3 |
| **C** | 1 1 1 |
| **D** | 3 3 3 |

1. **Predict the output of following C++ program**

|  |
| --- |
| *#include <iostream>*  *using namespace std;*    *class A*  *{*  *private:*  *int x;*  *public:*  *A(int \_x)  {  x = \_x; }*  *int get()  { return x; }*  *};*    *class B*  *{*  *static A a;*  *public:*  *static int get()*  *{  return a.get(); }*  *};*    *int main(void)*  *{*  *B b;*  *cout << b.get();*  *return 0;*  *}* |

|  |  |
| --- | --- |
| **A** | 0 |
| **B** | Compiler Error: Undefined reference B::a |
| **C** | Compiler Error: Cannot access static a |
| **D** | Compiler Error: multiple functions with same name get() |

1. **Predict the output of following C++ program**

|  |
| --- |
| *#include <iostream>*  *using namespace std;*  *int main()*  *{*  *const char\* p = "12345";*  *const char \*\*q = &p;*  *\*q = "abcde";*  *const char \*s = ++p;*  *p = "XYZWVU";*  *cout << \*++s;*  *return 0;*  *}* |

|  |  |
| --- | --- |
| **A** | Compiler Error |
| **B** | c |
| **C** | **B** |
| **D** | Garbage value |

1. **Predict the output of following C++ program**

|  |
| --- |
| *#include <iostream>*  *using namespace std;*  *class Point*  *{*  *int x, y;*  *public:*  *Point(int i = 0, int j =0)*  *{ x = i; y = j;  }*  *int getX() const { return x; }*  *int getY() {return y;}*  *};*    *int main()*  *{*  *const Point t;*  *cout << t.getX() << " ";*  *cout << t.gety();*  *return 0;*  *}* |

|  |  |
| --- | --- |
| **A** | Garbage Values |
| **B** | 0 0 |
| **C** | Compiler Error in line cout << t.getX() << " "; |
| **D** | Compiler Error in line cout << t.gety(); |

**Key**:

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| b | a | c | b | c | c | c | c | c | b |
|  |  |  |  |  |  |  |  |  |  |
| 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 |
| c | b | b | a | b | b | b | b | b | d |