**Quiz 02**

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 | |

* B

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 | |  |  | |

* **C**

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 |

* **A**

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 |

* B

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 |

* **B**

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

|  |
| --- |
|  |

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

* A

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 |

* B

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 |

* B

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 |

* B

1. **A is a friend class of B and B is a friend class of C, which is NOT TRUE**

|  |  |
| --- | --- |
| **A** | All of the members of the A have access to the private members of B. |
| **B** | A has no direct access to \*this pointer of B’s objects |
| **C** | B is not a friend class of A. |
| **D** | A is a friend class of C. |

* D

**Key**:

1 2 3 4 5 6 7 8 9 10

c c b b b b b b b d

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |