**Friend Functions And Classes:**

**Writer:** Sayak Haldar

**1. Which rule will not affect the friend function?**

a) private and protected members of a class cannot be accessed from outside

b) private and protected member can be accessed anywhere

c) protected member can be accessed anywhere

d) none of the mentioned

**Answer: a**

Explanation: Friend is used to access private and protected members of a class from outside the same class.

**2. Which keyword is used to declare the friend function?**

a) firend

b) friend

c) classfriend

d) myfriend

**Answer: b**

**3. What is the syntax of friend Class?**

a) friend class1 Class2;

b) friend class;

c) friend class

d) none of the mentioned

**Answer: d)**

The syntax is the following:

|  |
| --- |
| **class Node**  **{**  **private:**  **int key;**  **Node \*next;**  **/\* Other members of Node Class \*/**  **public**  **friend class LinkedList; // Now class  LinkedList can**  **// access private members of Node**  **};** |
|  |

Here, LinkedList class is declared as the friend class of Node

**4. What is the output of this program?**

#include <iostream>

using namespace std;

class Box

{

double width;

public:

friend void printWidth( Box box );

void setWidth( double wid );

};

void Box::setWidth( double wid )

{

width = wid;

}

void printWidth( Box box )

{

box.width = box.width \* 2;

cout << "Width of box : " << box.width << endl;

}

int main( )

{

Box box;

box.setWidth(10.0);

printWidth( box );

return 0;

}

a) 40

b) 5

c) 10

d) 20

**Answer: d**

Explanation: We are using the friend function for printwidth and multiplied the width value by 2, So

we got the output as 20

**Output:**

$ g++ friend.cpp

$ a.out

20

**5. What is the output of this program?**

#include <iostream>

using namespace std;

class sample

{

int width, height;

public:

void set\_values (int, int);

int area () {return (width \* height);}

friend sample duplicate (sample);

};

void sample::set\_values (int a, int b)

{

width = a;

height = b;

}

**//this is the friend function**

sample duplicate (sample rectparam)

{

**//creating duplicate using friend function**

**//now, this is just example**

**//In general, It is done using Copy constructor**

**//however, here in duplicate we are multiplying width and height by 2**

sample rectres;

rectres.width = rectparam.width \* 2;

rectres.height = rectparam.height \* 2;

return (rectres);

}

int main ()

{

sample rect, rectb;

rect.set\_values (2, 3);

rectb = duplicate (rect);

cout << rectb.area();

return 0;

}

a) 20

b) 16

c) 24

d) None of the mentioned

**Answer: c**

Explanation: In this program, we are using the friend function for duplicate function and calculating the area of the rectangle.

**Output:**

$ g++ friend1.cpp

$ a.out

24

**6. What is the output of this program?**

#include <iostream>

using namespace std;

class sample;

class sample1

{

int width, height;

public:

int area ()

{

return (width \* height);}

void convert (sample a);

};

class sample

{

private:

int side;

public:

void set\_side (int a)

{

side = a;

}

**//sample1 is a friend function of sample**

friend class sample1;

};

void sample1::convert (sample a)

{

**//Since, sample1 is a friend class of sample, it’s function can access sample’s private members’ value**

width = a.side;

height = a.side;

}

int main ()

{

sample sqr;

sample1 rect;

sqr.set\_side(6);

rect.convert(sqr);

cout << rect.area();

return 0;

}

a) 24

b) 35

c) 16

d) 36

**Answer: d**

Explanation: In this program, we are using the friend for the class and calculating the area of the square.

Output:

$ g++ friend2.cpp

$ a.out

36

**7. What is the output of this program?**

**#include <iostream>**

**using namespace std;**

**class base**

**{**

**int val1, val2;**

**public:**

**int get()**

**{**

**val1 = 100;**

**val2 = 300;**

**}**

**friend float mean(base ob);**

**};**

**float mean(base ob)**

**{**

**return float(ob.val1 + ob.val2) / 2;**

**}**

**int main()**

**{**

**base obj;**

**obj.get();**

**cout << mean(obj);**

**return 0;**

**}**

a) 200

b) 150

c) 100

d) 300

**Answer: a**

Explanation: In this program, We are finding the mean value by declaring the function mean as a friend of class base.

Output:

$ g++ friend3.cpp

$ a.out

200

**8. What is the output of this program?**

#include <iostream>

using namespace std;

class sample

{

private:

int a, b;

public:

void test()

{

a = 100;

b = 200;

}

friend int compute(sample e1);

};

int compute(sample e1)

{

return int(e1.a + e1.b) - 5;

}

int main()

{

sample e;

e.test();

cout << compute(e);

return 0;

}

a) 100

b) 200

c) 300

d) 295

**Answer) d) 295**

**9. Pick out the correct statement.**

a) A friend function may be a member of another class

b) A friend function may not be a member of another class

c) A friend function may or may not be a member of another class

d) None of the mentioned

**Answer: c**

**10. Where does keyword ‘friend’ should be placed?**

a) function declaration

b) function definition

c) main function

d) none of the mentioned

**Answer: a**

Explanation: The keyword friend is placed only in the function declaration of the friend function and not in the function definition because it is used to access the member of a class.