22F-3298\_QasimNaveed

/\*Task-01 : \*/

#include<iostream>

using namespace std;

class base

{

public:

int x;

protected:

int y;

private:

int z;

};

class publicDerived : public base

{

public:

publicDerived()

{

cout << "x is public" << endl;

cout << "y is protected" << endl;

cout << " z is not accessible from publicDerived" << endl;

}

};

class protectedDerived : protected base

{

public:

protectedDerived()

{

cout << " \nx is protected" << endl;

cout << " y is protected" << endl;

cout << " z is not accessible from protectedDerived" << endl;

}

};

class privateDerived : private base

{

public:

privateDerived()

{

cout << "\n x is private" << endl;

cout << " y is private" << endl;

cout << "z is not accessible from privateDerived" << endl;

}

};

int main()

{

}

/\* Task-02 : \*/

#include<iostream>

using namespace std;

class person

{

public:

person()

{

cout << "\nPerson constructor called \n";

};

~person()

{

cout << "\n Person Destructor called\n";

};

};

class insuree : public person

{

public:

insuree()

{

cout << "\n insuree constructor called\n";

}

~insuree()

{

cout << "\n insuree destructor called\n";

}

};

class admin : public person

{

public:

admin()

{

cout << "\n admin constructor called\n";

}

~admin()

{

cout << "\n admin destructr called\n";

}

};

class expert : public person

{

public:

expert()

{

cout << "\n expert constructor called\n";

}

~expert()

{

cout << "\n expert destructor called\n";

}

};

class thirdparty : public insuree

{

public:

thirdparty()

{

cout << "\n thirdarty constructor called\n";

}

~thirdparty()

{

cout << "\nthirdparty destructor called\n";

}

};

class customer : public insuree

{

public:

customer()

{

cout << "\n cstomer constructor called\n";

}

~customer()

{

cout << "\n customer destructor called\n";

}

};

int main()

{

thirdparty obj1;

cout << endl;

customer obj2;

cout << endl;

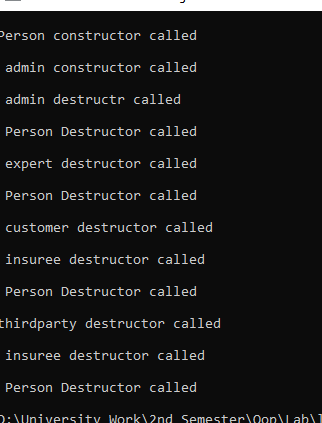
expert obj3;

cout << endl;

admin obj4;

return 0;

}



/\*Task-03 : \*/

#include<iostream>

using namespace std;

class Superclass

{

public:

Superclass(int x)

{

cout << "Superclass called" << endl;

}

};

class ClassA : public Superclass

{

public:

ClassA(int x) :Superclass(x)

{

cout << "ClassA called" << endl;

}

};

class ClassB : public Superclass

{

public:

ClassB(int x) :Superclass(x)

{

cout << "ClassB called" << endl;

}

};

class ClassC : public ClassA, public ClassB

{

public:

ClassC(int x) :ClassB(x), ClassA(x)

{

cout << "ClassC called" << endl;

}

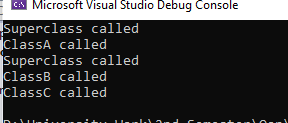
};

int main()

{

ClassC classC(30);

}



/\*Task-04\*/

#include<iostream>

using namespace std;

class card

{

private:

int num;

int suit;

char suits[4] = { '\6','\5','\3','\4'};

string numbers[13] = {"A","2","3","4","5","6","7","8","9","10","J","Q","K" };

public:

card(int num,int suit)

{

this->num = num;

this->suit = suit;

};

void display()

{

cout<< numbers[num] << " of " << suits[suit] << "\t";

}

};

int main()

{

for (int s = 0; s < 4; s++)

{

for (int n = 0; n < 13; n++)

{

card deck(n, s);

deck.display();

}

cout << endl;

}

}

