OPP LAB TASK

**4.1**

**Source Code:**

#include <iostream> #include<conio.h> using namespace std;

class BaseClass{ int num1; public:

void setInt(int n); int getInt();

};

class DerivedClass : public BaseClass{ int num2;

public:

void setJ(int n); int mul();

};

void BaseClass:: setInt(int n)

{

num1 = n;

}

int BaseClass::getInt()

{

return num1;

}

void DerivedClass::setJ(int n)

{

num2 = n;

}

int DerivedClass::mul()

{

cout<<"Result=";

return (num2 \* getInt());

}

int main()

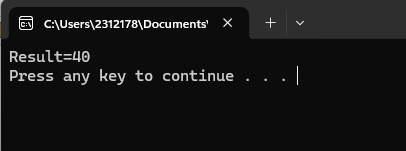
{

DerivedClass ob; ob.setInt(10); ob.setJ(4);

cout<<ob.mul()<<endl; system ("pause"); return 0;

}

**OutPut:**



**4.2**

**Source Code:**

#include<iostream>

using namespace std;

class Counter

{

protected:

unsigned int count; public:

Counter() : count(0)

{ }

Counter(int c) : count(c)

{ }

unsigned int get\_count() const

{ return count; } Counter operator ++ ()

{ return Counter(++count); }

};

class CountDn : public Counter

{

public:

Counter operator -- ()

{

return Counter(--count);

}

};

int main()

{

CountDn c1;

cout << "\nc1=" << c1.get\_count();

++c1; ++c1; ++c1;

cout << "\nc1=" << c1.get\_count();

for (int i = 0; i < 3; ++i)

++c1;

cout << "\nc1=" << c1.get\_count();

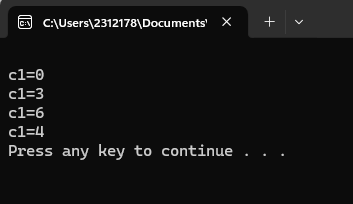
--c1; --c1;

cout << "\nc1=" << c1.get\_count(); cout << endl;

system("pause"); return 0;

}

**OUTPUT:**



**4.3**

**Source Code:** #include<iostream> #include<conio.h> using namespace std; class Stack { protected:

enum { MAX = 3 }; int st[MAX];

int top;

//NOTE: can't be private

//size of stack array

//stack: array of integers

//index to top of stack

public:

Stack() //constructor

{

top = -1;

}

void push(int var)

{

st[++top] = var; //put number on stack

}

int pop()

{

return st[top--]; //take number off stack

}

};

class Stack2 : public Stack { public:

void push(int var)

{

if(top >= MAX-1)

{

//put number on stack

//error if stack full

cout << "\nError: stack is full";

//exit(1);

}

Stack::push(var);

}

//call push() in Stack class

int pop()

{

//take number off stack

if(top < 0) //error if stack empty

{

cout << "\nError: stack is empty\n";

//exit(1);

}

return Stack::pop();

}

//call pop() in Stack class

};

int main()

{

Stack2 s1;

int top = -1; // add this line s1.push(11);

//push some values onto stack

s1.push(22);

s1.push(33);

cout << endl << s1.pop();

//pop some values from stack

cout << endl << s1.pop();

cout << endl << s1.pop();

if (top >= 0)

cout << endl << s1.pop(); else

cout << endl << "Error: stack is empty\n"; cout << endl;

system("pause");

//getch(); std::cin.ignore();

std::cin.get(); return 0;

}

**Output:**

