Name: Parab Ravindra Swami Roll no.: 194 Class: FYBCA

Div: B Practical :1 Write a program to check whether is even or odd using if-else

#include<iostream>

using namespace std;

int main ()

{

int number;

cout<<"Enter a number:";

cin>>number;

if(number%2==0)

cout<<number<<":Even";

else

cout<<number<<":Odd";

return 0;

}

Output:

Enter a number:2

2: Even

Name: Parab Ravindra Swami Roll no.: 194 Class: FYBCA

Div: B Practical :2Write a program to calculate the sum and average of three numbers using control structure.

#include<iostream>

using namespace std;

int main()

{

float num1,num2,num3,sum,average;

cout<<"Enter the first number:";

cin>>num1;

cout<<"Enter the second number:";

cin>>num2;

cout<<"Enter the third number:";

cin>>num3;

if(num1>=0 && num2>=0 &&num3>=0)

{

sum=num1+num2+num3;

average+sum/3;

cout<<"The sum of the number is:"<<sum<<endl;

cout<<"The average of the number is:"<<average<<endl;

}else{

cout<<"Error:Please Enter Only Non-Negative Number"<<endl;

}

return 0;

}

Output:

Enter the first number:23

Enter the second number:16

Enter the third number:18

The sum of the number is:57

The average of the number is:1.4013e-045

Name: Parab Ravindra Swami Roll no.: 194 Class: FYBCA

Div: B Practical : 3Write a program to calculate the sum, difference, product, and quotient of two integers.

#include<iostream>

using namespace std;

int main()

{

int num1,num2;

cout<<"Enter the First Integer:";

cin>>num1;

cout<<"Enter the second Integer:";

cin>>num2;

int sum=num1+num2;

int difference=num1-num2;

int product=num1\*num2;

if(num2!=0)

{

double qoutient=static\_cast<double>(num1)/num2;

cout<<"sum:"<<sum<<endl;

cout<<"difference:"<<difference<<endl;

cout<<"product:"<<qoutient<<endl;

}else{

cout<<"Division by zero is not allowed!"<<endl;

}

return 0;

}

Output:

Enter the First Integer:12

Enter the second Integer:12

sum:24

difference:0

product:1

Name: Parab Ravindra Swami Roll no.: 194 Class: FYBCA

Div: B Practical : 4Write a program to demonstrate encapsulation using of class.

#include<iostream>

using namespace std;

class Employee{

private:

int empld;

string empname;

float empsalary;

public:

void setempld(int id){

empld=id;

}

void setempname(string name){

empname=name;

}

void setempsalary(float salary){

empsalary=salary;

}

int getempld ()

{

return empld;

}

string getempname()

{

return empname;

}

float getempsalary()

{

return empsalary;

}

};

int main(){

Employee emp;

emp.setempld(194);

emp.setempname("Krishna");

emp.setempsalary(120000);

cout<<"Employee ID:"<<emp.getempld()<<endl;

cout<<"Employee Name:"<< emp.getempname()<<endl;

cout<<"Employee Salary:"<<emp.getempsalary()<<endl;

return 0;

}

Name: Om Ramakant Parvate Roll.no: 168 Class: FYBCA Div: B

Practical-5: Input/ Output in C++ (Cin, Cout, iostream)

#include<iostream>

using namespace std;

int main()

{

int i,n,a,prime=0;

cout<<"\nEnter the number:";

cin>>i;

for(n=2;n<i;n++)

{

if(i%n==0)

{

prime++;

}

}

if(prime==0)

{

cout<<"\nThis number is Prime";

}

else

{

cout<<"\nThis number is Composite";

}

return 0;

}

Output:

Enter the number:5

This number is Prime

Name: Om Ramakant Parvate Roll.no: 168 Class: FYBCA Div: B

Practical-6: Data types and Operators (Arithmetic, Relational, Logical, Bitwise)

#include <iostream>

using namespace std;

int main() {

int x = 10, y = 9, z = 5, w = 7;

cout << ++x << endl;

cout << --y << endl;

cout << +z << endl;

cout << -w << endl;

cout << -(-w) << endl;

int a = 10, b = 3;

int sum = a + b;

int difference = a - b;

int product = a \* b;

int quotient = a / b;

int remainder = a % b;

cout << "Sum: " << sum << endl;

cout << "Difference: " << difference << endl;

cout << "Product: " << product << endl;

cout << "Quotient: " << quotient << endl;

cout << "Remainder: " << remainder << endl;

return 0;

}

Output:

11

8

5

-7

7

Sum: 13

Difference: 7

Product: 30

Quotient: 3

Remainder: 1

Name: Om Ramakant Parvate Roll.no: 168 Class: FYBCA Div: B

Practical-7: . Write a program to demonstrate the use of different types of constructorsand a destructor in a class.

#include <iostream>

using namespace std;

class MyClass {

private:

int value;

public:

MyClass() {

value = 0;

cout << "Default constructor called. Value initialized to " << value << endl;

}

MyClass(int v) {

value = v;

cout << "Parameterized constructor called. Value initialized to " << value << endl;

}

MyClass(const MyClass &obj) {

value = obj.value;

cout << "Copy constructor called. Copied value: " << value << endl;

}

~MyClass() {

cout << "Destructor called for object with value: " << value << endl;

}

void displayValue() {

cout << "Value: " << value << endl;

}

};

int main() {

MyClass obj1;

obj1.displayValue();

MyClass obj2(200);

obj2.displayValue();

MyClass obj3 = obj2;

obj3.displayValue();

return 0;

}

Output:

Default constructor called. Value initialized to 0

Value: 0

Parameterized constructor called. Value initialized to 200

Value: 200

Copy constructor called. Copied value: 200

Value: 200

Destructor called for object with value: 200

Destructor called for object with value: 200

Destructor called for object with value: 0

Name: Om Ramakant Parvate Roll.no: 168 Class: FYBCA Div: B

Practical-8: Multiple Inheritance is a feature of C++ where a class can inherit from more than one classes. i.e one sub class is inherited from more than one base classes.

#include <iostream>

using namespace std;

class Vehicle {

public:

Vehicle()

{

cout << "This is a Vehicle" << endl;

}

};

class FourWheeler {

public:

FourWheeler()

{

cout << "This is a 2 wheeler Vehicle" << endl;

}

};

class Car: public Vehicle, public FourWheeler {

};

int main()

{

Car obj;

return 0;

}

Output:

This is a Vehicle

This is a 2 wheeler Vehicle

Name: Om Ramakant Parvate Roll.no: 168 Class: FYBCA Div: B

Practical-9: Function and its Components (SINGLE LEVEL)

#include <iostream>

using namespace std;

class Animal {

public:

void playing() {

cout << "I can playing!" << endl;

}

void run() {

cout << "I can run!" << endl;

}

};

class Dog : public Animal {

public:

void bark() {

cout << "I can bark! Woof woof!!" << endl;

}

};

int main() {

Dog dog1;

dog1.eat();

dog1.sleep();

dog1.bark();

return 0;

}

Output:

I can playing!

I can run!

I can bark! Woof woof!!

Name: Om Ramakant Parvate Roll.no:168 Class:FY BCA

Div:B

Practical:11-Unary Overloading;Defination

#include <iostream>

using namespace std;

class Distance {

public:

int feet, inch;

Distance(int f, int i)

{

this->feet = f;

this->inch = i;

}

void operator-()

{

feet--;

inch--;

cout << "\nFeet & Inches(Decrement): " <<

feet << "'" << inch;

}

};

int main()

{

Distance d1(10, 20);

-d1;

return 0;

}

Output:

Feet & Inches(Decrement): 9'19

Name: Om Ramakant Parvate Roll.no:168 Class:FY BCA

Div:B

Practical:12-Public, Privet, Protected

#include<iostream>

using namespace std;

class Complex

{

private:

int real,img;

public:

void read()

{

cout<<"Enter real & img part: ";

cin>>real>>img;

}

Complex operator+(Complex a)

{

Complex t;

t.real=real+a.real;

t.img=img+a.img;

return t;

}

void display()

{

cout<<"\n"<<real<<"+"<<img<<"i";

}

};

int main()

{

Complex c1,c2,c3;

c1.read();

c2.read();

c3=c1+c2;

c1.display();

c2.display();

cout<<"\nAddition:- ";

c3.display();

}

Output:

Enter real & img part: 1

3

Enter real & img part: 4

5

1+3i

4+5i

Addition:-

5+8i