Assignment 1 - Demonstrate the use of Control statements

1. If Statement:

#include<iostream.h>

#include<conio.h>

void main()

{

clrscr();

int i=10;

if(i<15)

{

cout<<”10 is less than 15”;

}

getch();

}

Output:

10 is less than 15

2. If-else statement: Even or Odd

#include<iostream.h>

#include<conio.h>

void main()

{

clrscr();

int num;

cout<<”\n Enter Number : \n” ;

cin>>num;

if(num%2==0)

{

cout<<”Even Number”;

}

else

{

cout<<”Odd Number”;

}

getch();

}

Output:

Enter No 14

Even Number

3. Sum and Average:

#include<iostream.h>

#include<conio.h>

void main()

{

clrscr();

int num1,num2,num3;

int sum, average;

cout<<”enter three numbers:”;

cin>>num1>>num2>>num3;

sum=num1+num2+num3;

average=sum/3;

cout<<”sum=”<<sum<<endl;

cout<<”average=”<<average<<endl;

getch();

}

Output:

enter three numbers: 10 20 15

sum=45

average=15

4. Prime or Composite Number

#include<iostream.h>

#include<conio.h>

void main()

{

clrscr();

int number,i,p=1;

cout<<”enter any number:”;

cin>>number;

i=2;

while(i<number)

{

if(number%i==0)

{

p=0;

break;

}

i++;

}

if(p==1)

{

cout<<”number is prime:”;

}

else

{

cout<<”number is composite”;

}

getch();

}

Output:- enter any number:1

number is prime or

enter any number:4

number is composite

5. Sum, Difference, Product and Quotient

#include<iostream.h>

#include<conio.h>

void main()

{

clrscr();

int x,y,sum,dif,pro,quot;

cout<<”enter first number:”;

cin>>x;

cout<<”enter second number:”;

cin>>y;

sum=x+y;

dif=x-y;

pro=x\*y;

quot=x/y;

cout<<”\nthe sum is:”<<sum<<endl;

cout<<”\nthe difference is:”<<dif<<endl;

cout<<”\nthe product is:”<<pro<<endl;

cout<<”\nthe quotient is:”<<quot<<endl;

getch();

}

Output:- Enter first number 2

Enter second number 3

The sum is 5

The difference is -1

The product is 6

The quotient is 0

Practical 2. Write Program to demonstrate use of Function Overloading

#include<iostream.h>

#include<conio.h>

class cal

{

public:

int add(int a, int b)

{

return a+b;

}

int add(int a, int b, int c)

{

return a+b+c;

}

};

int main()

{

clrscr();

cal c;

cout<<c.add(10,20)<<endl;

cout<<c.add(12,20,23);

getch();

}

Output:- 30 55

Practical 3.Write a program to demonstrate encapsulation using of class:

#include<iostream.h>

#include<conio.h>

class circle

{

private:

float area;

float radius;

public:

void getrRadius()

{

cout << "Enter radius\n";

cin >> radius;

}

void findArea()

{

area = 3.14 \* radius \* radius;

cout << "Area of circle=" << area;

}

};

int main()

{

clrscr();

circle cir;

cir.getRadius();

cir.findArea();

getch();

}

Output:- Enter radius

2

Area of circle=12.56

Practical 4.Write a program to demonstrate use constructors and Destructor

#include<iostream.h>

#include<conio.h>

class student

{

int rno;

char name[50];

public:

student()

{

cout << "Enter the RollNo:";

cin >> rno;

cout << "Enter the Name:";

cin >> name;

}

void display()

{

cout<<rno << "\t" <<name;

}

};

int main()

{

clrscr();

student s;

s.display();

getch();

return 0;

}

Output:- Default constructor is called

Name of current object: student

Age of current object: 12

 Parameterised constructor:-

#include <iostream.>

#include <string.h>

#include<conio.h>

class Student

{

private:

char name[20];

int age;

public:

Student(char\* n, int a)

{

strcpy(name, n);

age = a;

}

void display()

{

cout << "Name: " << name << endl;

cout << "Age: " << age << endl;

}

};

int main()

{

clrscr();

Student student1("John", 20);

student1.display();

Student student2("demo", 25);

student2.display();

getch();

return 0;

}

Output:- Name: John Age: 20

Name: demo Age: 25

 Destructor:-

#include <iostream.h>

#include <conio.h>

#include <string.h>

class Student

{

char name[20];

int age;

public:

Student(char\* n, int a)

{

strcpy(name, n);

age = a;

}

~Student()

{

cout << "Destructor called for " << name << endl;

delete[] name;

}

void displayInfo()

{

cout << "Name: " << name << endl;

cout << "Age: " << age << endl;

}

};

int main()

{

clrscr();

Student\* s = new Student("John ", 20);

s->displayInfo();

delete s;

getch();

return 0;

}

Output:- Name: John

Age: 20

Destructor called for john

Practical 5. Write a program to demonstrate single inheritance.

#include <iostream.h>

#include <conio.h>

class base

{

public:

int x;

void getdata()

{

cout << "Enter the value of x = ";

cin >> x;

}

};

class derive : public base

{

private:

int y;

public:

void readdata()

{

cout << "Enter the value of y = ";

cin >> y;

}

void product()

{

cout << "Product = " << x \* y;

}

};

int main()

{

clrscr();

derive a;

a.getdata();

a.readdata();

a.product();

getch();

return 0;

}

Output

Enter the value of x = 3

Enter the value of y = 4

Product = 12

Practical 6. Write a program to demonstrate multiple inheritances.

#include <iostream.h>

#include <conio.h>

class student\_detail

{

protected:

int rno, sum , i, marks[5];

public:

void detail()

{

cout << " Enter the Roll No: " << endl;

cin >> rno;

cout << " Enter the marks of five subjects " << endl;

for (i = 0; i < 5; i++)

{

cin >> marks[i];

}

for ( i = 0; i < 5; i++)

{

sum = sum + marks[i];

}

}

};

class sports\_mark

{

protected:

int s\_mark;

public:

void get\_mark()

{

cout << "\n Enter the sports mark: ";

cin >> s\_mark;

}

};

class result: public student\_detail, public sports\_mark

{

int tot, avg;

public:

void disp ()

{

tot = sum + s\_mark;

avg = tot / 6;

cout << " Roll No: " << rno << “ Total: " << tot << endl;

cout << " Average Marks: " << avg;

}

};

int main ()

{

clrscr();

result obj;

obj.detail();

obj.get\_mark();

obj.disp();

getch();

}

Output:- Enter the Roll No:

12

Enter the marks of five subjects

10

203

0

40

50

Enter the sports mark: 20

Roll No: 12 Total: 140

Average Marks: 23

Practical 7 .Write a program to demonstrate use of operator overloading using friend function.

#include<iostream.h>

#include<conio.h>

class complex

{

private:

int real;

int img;

public:

complex (int r = 0, int i = 0)

{

real = r;

img = i;

}

void Display ()

{

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

}

friend complex operator + (complex c1, complex c2);

};

complex operator + (complex c1,complex c2)

{

complex temp;

temp.real = c1.real + c2.real;

temp.img = c1.img + c2.img;

return temp;

}

int main ()

{

clrscr();

complex c1(5, 3), c2(10, 5), c3;

c1.Display();

cout << " + ";

c2.Display();

cout << " = ";

c3 = c1 + c2;

c3.Display();

getch();

}

Output:- 5+i3 + 10+i5 = 15+i8

8. Write a program to demonstrate use of friend function. Friend function declaration

#include <iostream.h>

#include <conio.h>

class base

{

private:

int private\_variable;

protected:

int protected\_variable;

public:

base()

{

private\_variable = 10;

protected\_variable = 99;

}

friend void friendFunction(base& obj);

};

void friendFunction(base& obj)

{

cout << "Private Variable: " << obj.private\_variable

<< endl;

cout << "Protected Variable: " << obj.protected\_variable;

}

int main()

{

clrscr();

base object1;

friendFunction(object1);

getch();

return 0;

}

Output:- private\_variable = 10;

protected\_variable = 99;

Practical 9. Write a program to demonstrate use of Virtual functions.

#include <iostream.h>

#include<conio.h>

class Animal

{

public:

virtual void sound()

{

cout << "The animal makes a sound." << endl;

}

};

class Dog : public Animal

{

public:

void sound()

{

cout << "The dog barks." << endl;

}

};

int main()

{

clrscr();

Dog myDog;

myDog.sound(); . getch();

return 0;

}

Output:-The dog barks

10. Write a program to demonstrate use of pointer. a) Write a program to demonstrate use of pointer to pointer.

#include<iostream.h>

#include<conio.h>

int main()

{

clrscr();

int age = 21;

int \*ptr1 = &age;

int \*\*ptr2 = &ptr1;

cout<<"Value stored in the age variable "<<age<<"\n";

cout<<"Value accessed using single pointer "<<\*ptr1<<"\n";

cout<<"Value accessed using double pointer "<<\*\*ptr2<<"\n";

getch();

return 0;

}

Output:- Value stored in the age variable 21

Value accessed using single pointer 21

Value accessed using double pointer 21

b) Write a program to demonstrate use of pointer to objects.

#include <iostream.h>

#include<conio.h>

class My\_Class

{

int num;

public:

void set\_number(int value)

{

num = value;

}

void show\_number();

};

void My\_Class::show\_number()

{

cout << num << "\n";

}

int main()

{

clrscr();

My\_Class object, \*p;

object.set\_number(1);

object.show\_number();

p = &object; p->show\_number();

getch();

return 0;

}

Output:- 1

1

C)Write a program to demonstrate use of pointer to function.

#include <iostream.h>

#include<conio.h>

int mult(int x, int y)

{

return x \* y;

}

int main()

{

clrscr();

int (\*funcptr)(int, int);

funcptr = mult;

int mul = funcptr(5, 7);

cout << "The value of the product is: " << mul << endl;

getch();

return 0;

}

Output:- The value of the product is: 35

Practical 11. Write a program to demonstrate use of Exception Handling.

#include <iostream.h>

#include<conio.h>

int main()

{

clrscr();

double numerator, denominator, divide;

cout << "Enter numerator: ";

cin >> numerator;

cout << "Enter denominator: ";

cin >> denominator;

try

{

if (denominator == 0)

throw 0;

divide = numerator / denominator;

cout << numerator << " / " << denominator << " = " << divide << endl;

}

catch (int num\_exception)

{

cout << "Error: Cannot divide by " << num\_exception << endl;

}

getch();

return 0;

}

Output:

Enter numerator:72

Enter denominator:0

Error: Cannot divide by 0

A constructor can be declared – 1. Inside the class

2. Outside the class (using scope resolution :: operator)

Example:-

#include <iostream.h>

class Person{

private: string name;

int age;

public:

Person()

{

cout<<"Default constructor is called"<<endl;

name = "student";

age = 12;

}

void display(){

cout<<"Name of current object: "<<name<<endl;

cout<<"Age of current object: "<<age<<endl;

}

};

int main()

{

Person obj;

obj.display();

return 0;

}

Output

Default constructor is called

Name of current object: student

Age of current object: 12