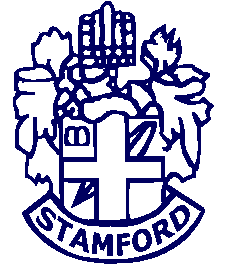
# Stamford University Bangladesh



***A Project Report***

***On***

***Overloading Overview Calculator.***

*Course Title: Object Oriented Programming Sessional.*

*Course code:CSI 218*

***Submitted to:***

*Name: Nahida Parvin.*

*Designation: Lecturer Department of CSE.*

*Stamford University Bangladesh*

***Submitted by.***

*Name:*

*Md.Jannatul Ferdous Rasel. (ID : CSE 05706898)*

*Md.Tutul hossain ( ID:CSE06007270)  
 Rakib Hassan ( ID:CSE06007262)*

*Batch No. 60(c).*

*Program: B.sc in CSE.*

*Stamford University Bangladesh. Date of Submission:10-04-2017*

***PROJECT DESCRIPTION***

A **scientific calculator** is a type of [electronic](https://en.wikipedia.org/wiki/Electronics) [calculator](https://en.wikipedia.org/wiki/Calculator), usually but not always handheld, designed to calculate problems in [science](https://en.wikipedia.org/wiki/Science), [engineering](https://en.wikipedia.org/wiki/Engineering), and [mathematics](https://en.wikipedia.org/wiki/Mathematics). They have almost completely replaced [slide rules](https://en.wikipedia.org/wiki/Slide_rule) in almost all traditional applications, and are widely used in both education and professional settings.

In certain contexts such as higher education, scientific calculators have been superseded by [graphing calculators](https://en.wikipedia.org/wiki/Graphing_calculator), which offer a superset of scientific calculator functionality along with the ability to graph input data and write and store programs for the device. There is also some overlap with the financial calculator market.

**Advantage:**1. Easily back into main function.  
2. Easily exit any switch.  
3. Sunation, multification and subtract Nth term variable it's user define.  
4. Any user can use it easily.  
Limitation:  
1. Run time is around 4 second.  
2. Arithmetic operation division use for only two variable divide.  
3. We can not use all Arithmetic operation together.  
4. Switch only perform integer value.

## Contents

**1.Arith  
2.Trigo  
3.Triangle  
4.Rectangle  
5.Trapezoid  
6.Ellipse  
7.Square  
8.Parallelogram  
9.Circle  
10.Sector  
11.Cube  
12.Sphere  
13.Cone**

**14.Torus  
15.Logarithmicfunction  
16.Powerfunction  
17.matrix  
18. factorial  
19. Factorial  
20.s\_digit  
21.Average  
22.Percentage  
23.Prime  
24.Series  
25.Converter  
26.Complex**

**PROJECT MAIN AND SUB SWITCH:**

1. Arithmetic Operations   
  
2. Trigonometric Functions  
  
3. Logarithmic Functions

4. Power Functions  
 **5. complex number  
  
6. Matrix**

**7. add digits**

**8. n!   
  
9. nCr & npr  
  
10. area calculate**

**11. Summation different type of series  
  
12. average the numbers**

**13. percentage the number**

**14. check the prime number**

**15. NUMBER SYSTERM CONVERTION**

**16. Exit**

## Functions

Modern scientific calculators generally have many more features than a standard four or five-function calculator, and the feature set differs between manufacturers and models; however, the defining features of a scientific calculator include:

**1 : Arithmetic Operations .  
 2 : Trigonometric Functions.  
 3 : Logarithmic Functions.  
 4 : Power Functions.  
 5 : complex number.  
 6 : matrix .  
 7 : add digits.  
 8 : n! .  
 9 : nCr & npr.  
 10 : area calculate.  
 11 : summation different type of series.  
 12 : average the numbers.  
 13 : percentage the number.  
 14 : check the prime number.  
 15 : NUMBER SYSTERM CONVERTION.  
 16 : Exit.**

## Uses

Scientific calculators are used widely in any situation where quick access to certain mathematical functions is needed, especially those that were once looked up in tables, such as trigonometric functions or logarithms; they are also used in situations requiring calculations of very large or very small numbers, as in some aspects of [astronomy](https://en.wikipedia.org/wiki/Astronomy), [physics](https://en.wikipedia.org/wiki/Physics), and [chemistry](https://en.wikipedia.org/wiki/Chemistry).

They are very often required for math classes from the junior high school level through college, and are generally either permitted or required on many [standardized tests](https://en.wikipedia.org/wiki/Standardized_test) covering math and science subjects; as a result, many are sold into educational markets to cover this demand, and some high-end models include features making it easier to translate a problem on a textbook page into calculator input, e.g. by providing a method to enter an entire problem in as it is written on the page using simple formatting tools.

## History

The first scientific calculator that included all of the basic ideas above was the programmable Hewlett-Packard [HP-9100A](https://en.wikipedia.org/wiki/Hewlett-Packard_9100A),[[1]](https://en.wikipedia.org/wiki/Scientific_calculator#cite_note-1) released in 1968, though the [Wang](https://en.wikipedia.org/wiki/Wang_Laboratories) LOCI-2 and the Mathatronics Mathatron had some features later identified with scientific calculator designs. The HP-9100 series was built entirely from discrete [transistor](https://en.wikipedia.org/wiki/Transistor) logic with no [integrated circuits](https://en.wikipedia.org/wiki/Integrated_circuit), and was one of the first uses of the [CORDIC](https://en.wikipedia.org/wiki/CORDIC) algorithm for trigonometric computation in a personal computing device, as well as the first calculator based on [Reverse Polish Notation](https://en.wikipedia.org/wiki/Reverse_Polish_notation) (RPN) entry. HP became closely identified with RPN calculators from then on, and even today some of their high-end calculators (particularly the long-lived [HP-12C](https://en.wikipedia.org/wiki/HP-12C) financial calculator and the [HP-48](https://en.wikipedia.org/wiki/HP-48) series of graphing calculators) still offer RPN as their default input mode due to having garnered a very large following.

The [HP-35](https://en.wikipedia.org/wiki/HP-35), introduced on February 1, 1972, was Hewlett-Packard's first [pocket calculator](https://en.wikipedia.org/wiki/Pocket_calculator) and the world's first handheld scientific calculator.[[2]](https://en.wikipedia.org/wiki/Scientific_calculator#cite_note-2) Like some of HP's desktop calculators it used RPN. Introduced at US$395, the HP-35 was available from 1972 to 1975.

**Challenges in the use of scientific calculators**

Current mathematics practice in Kenya is textbook driven. This means that it is teacher- centered and consists mainly of transmission of knowledge from the teacher to students. This completely ignores the fact that students are of varied abilities with different backgrounds and fails to take into account individual differences. This results in the learners becoming passive recipients of knowledge and consequently develops negative attitudes towards the subject. Thus, mathematics is reduced to memorization of procedures, facts, formulae and algorithms. In line with this, the study sought to find out the challenges learners face in the process of using scientific calculators in a mathematics class during learning sessions and in mathematics examinations time.



**APPENDEX**

#include<iostream>

#include<stdio.h>

#include<string.h>

#include <cstdlib>

#include<math.h>

#include<conio.h>

#define MAX 1000

#define PI 3.14159265

using namespace std;

class Arith

{

public:

float x1,y1;

float add();

float sub();

float mul();

float div();

};

float Arith::add()

{

int n,i;

cout<<"\tEnter the Total Number you want to Sum: ";

cin>>n;

cout<<endl;

float a[n],ans=0;

for(i=1;i<=n;i++)

{

cout<<"\tEnter the "<<i<<" no Number: ";

cin>>a[i];

ans=ans+a[i];

}

return ans;

}

float Arith::sub()

{

int n,i,x21,an1;

cout<<"\tEnter the Total Number you want to subtract: ";

cin>>n;

cout<<endl;

cout<<"\tEnter the 1 no number: ";

cin>>x21;

float a[n],ans=0;

for(i=1;i<=n-1;i++)

{

cout<<"\tEnter the "<<i+1<<" no Number: ";

cin>>a[i];

ans=ans+a[i];

an1=x21-ans;

}

return an1;

}

float Arith::mul()

{

int n,i;

cout<<"\tEnter the Total Number you want to Multifly: ";

cin>>n;

cout<<endl;

float a[n],ans=1;

for(i=1;i<=n;i++)

{

cout<<"\tEnter the "<<i<<" no Number: ";

cin>>a[i];

ans=ans\*a[i];

}

return ans;

}

float Arith::div()

{

cout<<"\tEnter the 1st value: ";

cin>>x1;

cout<<"\n\tEnter the 2nd value: ";

cin>>y1;

return(x1/y1);

}

class Trigo

{

public:

int n1;

float n,ans,ans1;

float Sin();

float Cos();

float Tan();

float Asin();

float Acos();

float Atan();

};

float Trigo::Sin()

{

cout<<"\tEnter the Angle: ";

cin>>n;

cout<<endl;

ans=sin(n\*PI/180);

cout<<"\tsin "<<n<<" Ans= "<<ans<<endl;

}

float Trigo::Cos()

{

cout<<"\tEnter the Angle: ";

cin>>n;

cout<<endl;

if(n==90)

{

ans=0;

cout<<"\tcos "<<n<<" ans= "<<ans<<endl;

}

else

{

ans=cos(n\*PI/180);

cout<<"\tcos "<<n<<" ans= "<<ans<<endl;

}

}

float Trigo::Tan()

{

cout<<"\tEnter the Angle: ";

cin>>n1;

cout<<endl;

ans=tan(n1\*PI/180);

if(n1%90==0)

{

cout<<"\ttan "<<n1<<" ans= Undefined"<<endl;

}

else

cout<<"\ttan "<<n1<<" ans= "<<ans<<endl;

}

float Trigo::Asin()

{

cout<<"\tEnter the value: ";

cin>>n;

cout<<endl;

ans1=asin(n);

ans=(ans1\*180)/PI;

cout<<"\tsin^-1 "<<n<<"= "<<ans<<endl;

}

float Trigo::Acos()

{

cout<<"\tEnter the value: ";

cin>>n;

cout<<endl;

ans1=acos(n);

ans=(ans1\*180)/PI;

cout<<"\tcos^-1 "<<n<<"= "<<ans<<endl;

}

float Trigo::Atan()

{

cout<<"\tEnter the value: ";

cin>>n;

cout<<endl;

ans1=atan(n);

ans=(ans1\*180)/PI;

cout<<"\ttan^-1 "<<n<<"= "<<ans<<endl;

}

class Iogarithmicfunction

{

public:

float n,ans,ans1;

float Log();

float Log10();

};

float Iogarithmicfunction::Log()

{

cout<<"\tEnter the value: ";

cin>>n;

cout<<endl;

ans=log(n);

cout<<"\tln Ans= "<<ans<<endl;

}

float Iogarithmicfunction::Log10()

{

cout<<"\tEnter the value: ";

cin>>n;

cout<<endl;

ans=log10(n);

cout<<"\tlog ans= "<<ans<<endl;

}

class Powerfunction

{

public:

float n,ans,x;

float Pow();

float Sqrt();

};

float Powerfunction::Pow()

{

cout<<"\tEnter the value of x: ";

cin>>x;

cout<<endl;

cout<<"\tEnter the value of number: ";

cin>>n;

cout<<endl;

ans=pow(x,n);

cout<<"\t x^n= "<<ans<<endl;

}

float Powerfunction::Sqrt()

{

cout<<"\tEnter the value: ";

cin>>n;

cout<<endl;

ans=sqrt(n);

cout<<"\tsqrt n= "<<ans<<endl;

}

class complex

{

public:

float a0,b0;

void read()

{

cout<<"\n\n\tEnter the REAL PART : ";

cin>>a0;

cout<<"\n\n\tEnter the IMAGINARY PART : ";

cin>>b0;

}

complex operator +(complex c2)

{

complex c3;

c3.a0=a0+c2.a0;

c3.b0=b0+c2.b0;

return c3;

}

complex operator -(complex c2)

{

complex c3;

c3.a0=a0-c2.a0;

c3.b0=b0-c2.b0;

return c3;

}

complex operator \*(complex c2)

{

complex c3;

c3.a0=(a0\*c2.a0)-(b0\*c2.b0);

c3.b0=(b0\*c2.a0)+(a0\*c2.b0);

return c3;

}

complex operator /(complex c2)

{

complex c3;

c3.a0=((a0\*c2.a0)+(b0\*c2.b0))/((c2.a0\*c2.a0)+(c2.b0\*c2.b0));

c3.b0=((b0\*c2.a0)-(a0\*c2.b0))/((c2.a0\*c2.a0)+(c2.b0\*c2.b0));

return c3;

}

void display()

{

cout<<a0<<"+"<<b0<<"i";

}

};

class matrix

{

public :

int a5[10][10],b5[10][10],c5[10][10],d5[10][10],e5[10][10],f5[10][10],x2,y2,i1,j1;

void transpose();

void sum2();

void diff();

void multifica1();

};

void matrix::sum2()

{

cout<< "\tEnter the rows: ";

cin>> x2;

cout<< "\tEnter the columns: ";

cin>> y2;

cout<< "\tEnter elements of first matrix: \n\n";

for (i1=1; i1<=x2; i1++)

{

for ( j1=1; j1<=y2; j1++)

{

cin>> a5[i1][j1];

}

}

cout<< "\tEnter elements of second matrix: \n\n";

for (i1=1; i1<=x2; i1++)

{

for (j1=1; j1<=y2; j1++)

{

cin>> c5[i1][j1];

}

}

cout<< "\tSum of Matrices 1 and 2 is: \n";

for (i1=1; i1<=x2; i1++)

{

for ( j1=1; j1<=y2; j1++)

{

e5[i1][j1]=a5[i1][j1]+c5[i1][j1];

cout<< e5[i1][j1] << " ";

}

cout<<endl;

}

}

void matrix::diff()

{

cout<< "\tEnter the rows: ";

cin>> x2;

cout<< "\tEnter the columns: ";

cin>> y2;

cout<< "\tEnter elements of first matrix: \n\n";

for (i1=1; i1<=x2; i1++)

{

for ( j1=1; j1<=y2; j1++)

{

cin>> a5[i1][j1];

}

}

cout<< "\tEnter elements of second matrix: \n\n";

for (i1=1; i1<=x2; i1++)

{

for (j1=1; j1<=y2; j1++)

{

cin>> c5[i1][j1];

}

}

cout<< "\tDifference of Matrices 1 and 2 (1-2) is: \n";

for (i1=1; i1<=x2; i1++)

{

for ( j1=1; j1<=y2; j1++)

{

f5[i1][j1] = a5[i1][j1]-c5[i1][j1];

cout<< f5[i1][j1] << " ";

}

cout<<endl;

}

}

void matrix::transpose()

{

cout<< "\tEnter the rows: ";

cin>> x2;

cout<< "\tEnter the columns: ";

cin>> y2;

cout<< "\tEnter elements of first matrix: \n\n";

for (i1=1; i1<=x2; i1++)

{

for ( j1=1; j1<=y2; j1++)

{

cin>> a5[i1][j1];

}

}

cout<< "\tEnter elements of second matrix: \n\n";

for (i1=1; i1<=x2; i1++)

{

for (j1=1; j1<=y2; j1++)

{

cin>> c5[i1][j1];

}

}

cout<< "\tTranspose of the matrix is: \n";

for ( i1=1; i1<=x2; i1++)

{

for ( j1=1; j1<=y2; j1++)

{

b5[i1][j1] = a5[j1][i1];

cout<< b5[i1][j1] << " ";

}

cout<<endl;

}

cout<< "\tTranspose of the second matrix is: \n";

for ( i1=1; i1<=x2; i1++)

{

for ( j1=1; j1<=y2; j1++)

{

d5[i1][j1] = c5[j1][i1];

cout<< b5[i1][j1] << " ";

}

cout<<endl;

}

}

void matrix::multifica1()

{

cout<< "\tEnter the rows: ";

cin>> x2;

cout<< "\tEnter the columns: ";

cin>> y2;

cout<< "\tEnter elements of first matrix: \n\n";

for (i1=1; i1<=x2; i1++)

{

for ( j1=1; j1<=y2; j1++)

{

cin>> a5[i1][j1];

}

}

cout<< "\tEnter elements of second matrix: \n\n";

for (i1=1; i1<=x2; i1++)

{

for (j1=1; j1<=y2; j1++)

{

cin>> c5[i1][j1];

}

}

cout<< "\tMultification of Matrices 1 and 2 is: \n";

for (i1=1; i1<=x2; i1++)

{

for ( j1=1; j1<=y2; j1++)

{

e5[i1][j1]=a5[i1][j1]\*c5[i1][j1];

cout<< e5[i1][j1] << " ";

}

cout<<endl;

}

}

class s\_digit

{

public:

int sum,num;

void input();

void calculate();

void display();

};

void s\_digit::input()

{

cout<<"\n\tEnter a natural no: ";

cin>>num;

}

void s\_digit::calculate()

{

int no,d;

sum=0;

no=num;

while(no>0)

{

d=no%10;

sum=sum+d;

no=no/10;

}

}

void s\_digit::display()

{

cout<<"\n\tThe given no is: "<<num<<endl;

cout<<"\n\tSum of the digits: "<<sum;

}

class factorial

{

public:

int f, n;

void fact();

void display();

};

void factorial::fact()

{

f=1;

cout<<"\n\tEnter a Number: ";

cin>>n;

for(int i=1;i<=n;i++)

f=f\*i;

}

void factorial::display()

{

cout<<"\n\tFactorial of "<<n<<" is: "<<f;

}

class Factorial{

public:

int f1,f2,f3, n,r;

void Factn();

void Factr();

void Factnr();

void display();

void display1();

};

void Factorial::Factn()

{

f1=1;

cout<<"\n\tEnter the value of N: ";

cin>>n;

for(int i=1;i<=n;i++)

f1=f1\*i;

}

void Factorial::Factr()

{

f2=1;

cout<<"\n\tEnter the value of R: ";

cin>>r;

for(int i=1;i<=r;i++)

f2=f2\*i;

}

void Factorial::Factnr()

{

f3=1;

for(int i=1;i<=n-r;i++)

f3=f3\*i;

}

void Factorial::display()

{

long ncr;

ncr=f1/(f2\*f3);

cout<<"\n\tnCr of "<<" is: "<<ncr<<endl;

}

void Factorial::display1()

{

long npr;

npr=(f1)/f3;

cout<<"\n\tnPr of "<<" is: "<<npr<<endl;

}

class Number

{

public:

float base,height,a2,b2,angle,radius,side,length,width;

void getNumber(void)

{

cout << "\tEnter base: ";

cin >> base;

}

void get1Number(void)

{

cout << "\tEnter height: ";

cin >> height;

}

void get2Number(void)

{

cout << "\tEnter a2: ";

cin >> a2;

}

void get3Number(void)

{

cout << "\tEnter b2: ";

cin >> b2;

}

void get4Number(void)

{

cout << "\tEnter angle: ";

cin >> angle;

}

void get5Number(void)

{

cout << "\tEnter radius: ";

cin >> radius;

}

void get6Number(void)

{

cout << "\tEnter side: ";

cin >> side;

}

void get7Number(void)

{

cout << "\tEnter length: ";

cin >> length;

}

void get8Number(void)

{

cout << "\tEnter width: ";

cin >> width;

}

float returnNumber(void)

{

return base;

}

float return1Number(void)

{

return height;

}

float return2Number(void)

{

return a2;

}

float return3Number(void)

{

return b2;

}

float return4Number(void)

{

return angle;

}

float return5Number(void)

{

return radius;

}

float return6Number(void)

{

return side;

}

float return7Number(void)

{

return length;

}

float return8Number(void)

{

return width;

}

};

class Triangle:public Number

{

public:

float getTriangle(void)

{

float base,Tri,height;

base=returnNumber();

height=return1Number();

Tri=(base\*height)/2;

return Tri;

}

};

class Rectangle:public Number

{

public:

float getRectangle(void)

{

float length,width,Rec;

length=return7Number();

width=return8Number();

Rec=length\*width;

return Rec;

}

};

class Trapezoid:public Number

{

public:

float getTrapezoid(void)

{

float b2,Tra,height,a2;

a2=return2Number();

b2=return3Number();

height=return1Number();

Tra=(height\*(a2+b2))/2;

return Tra;

}

};

class Ellipse:public Number

{

public:

float getEllipse(void)

{

float a2,Elli,b2;

a2=return2Number();

b2=return3Number();

Elli=a2\*b2\*PI;

return Elli;

}

};

class Square:public Number

{

public:

float getSquare(void)

{

float side,Squa;

side=return6Number();

Squa=side\*side;

return Squa;

}

};

class Parallelogram:public Number

{

public:

float getParallelogram(void)

{

float base,Paral,height;

base=returnNumber();

height=return1Number();

Paral=(base\*height);

return Paral;

}

};

class Circle:public Number

{

public:

float getCircle(void)

{

float radius,Cir;

radius=return5Number();

Cir=PI\*radius\*radius;

return Cir;

}

};

class Sector:public Number

{

public:

float getSector(void)

{

float radius,Sec,angle;

radius=return5Number();

angle=return4Number();

Sec=(radius\*radius\*angle)/2;

return Sec;

}

};

class Cube:public Number

{

public:

float getCube(void)

{

float radius,Cub;

radius=return5Number();

Cub=6\*radius\*radius;

return Cub;

}

};

class Sphere:public Number

{

public:

float getSphere(void)

{

float radius,Sph;

radius=return5Number();

Sph=4\*PI\*radius\*radius;

return Sph;

}

};

class Cone:public Number

{

public:

float getCone(void)

{

float radius,Con,height;

radius=return5Number();

a2=return2Number();

Con=PI\*radius\*a2;

return Con;

}

};

class Torus:public Number

{

public:

float getTorus(void)

{

float b2,Tor,a2;

b2=return3Number();

a2=return2Number();

Tor=PI\*PI\*(b2\*b2-a2\*a2);

return Tor;

}

};

class Series

{

public:

float n1,sum1,i,term1,a2,r1,d1;

float HPseries();

float GPseries();

float APseries();

};

float Series::HPseries()

{

cout<<"1+1/2+1/3+....+1/n1"<<endl;

cout<<"\tEnter the value of n1: ";

cin>>n1;

cout<<endl;

sum1=0;

for(i=1;i<=n1;i++)

{

term1=1/i;

sum1=sum1+term1;

}

cout<<"\tthe sum of Harmonic Progression(HP) series is: "<<sum1<<endl;

}

float Series::GPseries()

{

cout<<"\tEnter the first number of the Geometric Progression (G.P) series: ";

cin>>a2;

cout<<endl;

cout<<"\tEnter the total numbers of the Geometric Progression (G.P) series: ";

cin>>n1;

cout<<endl;

cout<<"\tEnter the common ratio of the Geometric Progression (G.P) series: ";

cin>>r1;

cout<<endl;

sum1=(a2\*(1-pow(r1,n1-1)))/(1-r1);

term1=a2\*(1-pow(r1,n1-1));

cout<<"\tterm1 of Geometric Progression (G.P) series: "<<term1<<endl;

cout<<"\tsum of Geometric Progression (G.P) series: "<<sum1<<endl;

}

float Series::APseries()

{

cout<<"\tEnter the first term value of the arithmetic progression series(A.P): ";

cin>>a2;

cout<<endl;

cout<<"\tEnter the total numbers of the arithmetic progression series (A.P): ";

cin>>n1;

cout<<endl;

cout<<"\tEnter the common difference the arithmetic progression series(A.P): ";

cin>>d1;

cout<<endl;

sum1=(n1\*(2\*a2+(n1-1)\*d1))/2;

term1=a2+(n1-1)\*d1;

cout<<"\tsum of arithmetic progression series (A.P): "<<sum1<<endl;

for(i=a2;i<=term1;i=i+d1)

{

if(i!=term1)

cout<<" + "<<i;

else

cout<<" = "<<sum1<<endl;

}

}

class Average

{

public:

float add1();

};

float Average::add1()

{

int n6,i6;

cout<<"\tEnter the Total Number you want to average: ";

cin>>n6;

cout<<endl;

float a6[n6],sum6=0,average;

cout<<"\tEnter the "<<n6<<" values"<<endl;

for(i6=1;i6<=n6;i6++)

{

cout<<"\tEnter the "<<i6<<" no Value:";

cin>>a6[i6];

sum6=sum6+a6[i6];

average=sum6/n6;

}

cout<<"\taverage= "<<average<<endl;

return average;

}

class Percentage

{

public:

float per();

};

float Percentage::per()

{

float percentage,n7,t;

cout<<"\tEnter the Number: ";

cin>>n7;

cout<<endl;

cout<<"\tEnter the total number: ";

cin>>t;

cout<<endl;

percentage=(n7/t)\*100;

cout<<"\tpercentagee= "<<percentage <<"%"<<endl;

return percentage;

}

class Prime

{

public:

float prime();

};

float Prime::prime()

{

int n8, i9;

bool isPrime = true;

cout << "\n\tEnter a positive integer: ";

cin >> n8;

for(i9 = 2; i9 <= n8 / 2; i9++)

{

if(n8 % i9 == 0)

{

isPrime = false;

break;

}

}

if (isPrime)

cout << "\n\tThis is a prime number";

else

cout << "\n\tThis is not a prime number";

return 0;

}

class convertor

{

public:

int DtoB();

int DtoO();

int DtoHD();

int BtoD();

int BtoO();

int BtoHD();

int OtoD();

int OtoB();

int OtoHD();

int HDtoD();

int HDtoB();

char HDtoO();

};

int convertor::DtoB()

{

int rem,decimal,binary=0,base=1;

cout<<"\n\tEnter a decimal number: ";

cin>>decimal;

while(decimal>0)

{

rem=decimal%2;

binary=binary+(rem\*base);

base=base\*10;

decimal=decimal/2;

}

cout<<"\n\tBinary of given number is: "<<binary;

}

int convertor::DtoO()

{

int rem,decimal,octal=0,base=1;

cout<<"\n\tEnter a decimal number: ";

cin>>decimal;

while(decimal>0)

{

rem=decimal%8;

octal=octal+(rem\*base);

base=base\*10;

decimal=decimal/8;

}

cout<<"\n\tOctal of given number is: "<<octal;

}

int convertor::DtoHD()

{

long int decnum, rem, quot;

char hexdecnum[100];

int i=1, j, temp;

cout<<"\n\tEnter any decimal number: ";

cin>>decnum;

quot=decnum;

while(quot!=0)

{

temp=quot%16;

if(temp<10)

{

temp=temp+48;

}

else

{

temp=temp+55;

}

hexdecnum[i++]=temp;

quot=quot/16;

}

cout<<"\n\tHexadecimal of given number is: ";

for(j=i-1; j>0; j--)

{

cout<<hexdecnum[j];

}

}

int convertor::BtoD()

{

int rem,i15=0,decimal=0,binery;

cout<<"\n\tEnter a binery number: ";

cin>>binery;

while(binery!=0)

{

rem=binery%10;

decimal+=rem\*pow(2,i15);

binery=binery/10;

i15++;

}

cout<<"\n\tDecimal of given number is: "<<decimal;

}

int convertor::BtoO()

{

int binaryNumber;

cout << "\n\tEnter a binary number: ";

cin >> binaryNumber;

int octalNumber = 0, decimalNumber = 0, i = 0;

while(binaryNumber != 0)

{

decimalNumber += (binaryNumber%10) \* pow(2,i);

++i;

binaryNumber/=10;

}

i = 1;

while (decimalNumber != 0)

{

octalNumber += (decimalNumber % 8) \* i;

decimalNumber /= 8;

i \*= 10;

}

cout << "\n\tOctal of given number is= " << octalNumber ;

}

int convertor::BtoHD()

{

char binnum[MAX], hexa[MAX];

int temp;

long int i=0,j=0;

cout<<"\n\tEnter Binary Number: ";

cin>>binnum;

while(binnum[i])

{

binnum[i] = binnum[i] -48;

++i;

}

--i;

while(i-2>=0)

{

temp = binnum[i-3] \*8 + binnum[i-2] \*4 + binnum[i-1] \*2 + binnum[i] ;

if(temp > 9)

hexa[j++] = temp + 55;

else

hexa[j++] = temp + 48;

i=i-4;

}

if(i ==1)

hexa[j] = binnum[i-1] \*2 + binnum[i] + 48 ;

else if(i==0)

hexa[j] = binnum[i] + 48 ;

else

--j;

cout<<"\n\tHexadecimal of given Number is: ";

while(j>=0)

{

cout<<hexa[j--];

}

}

int convertor::OtoD()

{

int rem,i15=0,octal,decimal=0;

cout<<"\n\tEnter A octal number: ";

cin>>octal;

while(octal!=0)

{

rem=octal%10;

decimal+=rem\*pow(8,i15);

octal=octal/10;

i15++;

}

cout<<"\n\tdecimal OF GIVEN NUMBER IS: "<<decimal;

}

int convertor::OtoB()

{

int octalNumber;

cout << "\n\tEnter an octal number: ";

cin >> octalNumber;

int decimalNumber = 0, i = 0;

long long binaryNumber = 0;

while(octalNumber != 0)

{

decimalNumber += (octalNumber%10) \* pow(8,i);

++i;

octalNumber/=10;

}

i = 1;

while (decimalNumber != 0)

{

binaryNumber += (decimalNumber % 2) \* i;

decimalNumber /= 2;

i \*= 10;

}

cout <<"\n\tBinary of given number is: " << binaryNumber;

}

int convertor::OtoHD()

{

int a[20], b[20], c[20], rev[20], h, i, j, k, l, x, fra, flag, rem, num1, num3;

float rem1, num2, num4, dno;

char s[20];

x = fra = flag = rem = 0;

rem1 = 0.0;

cout<<"\n\tEnter a octal number: ";

cin>>s;

for(i=0,j=0,k=0; i<strlen(s); i++)

{

if(s[i]=='.')

{

flag=1;

}

else if(flag==0)

{

a[j++]=s[i]-48;

}

else if(flag==1)

{

b[k++]=s[i]-48;

}

}

x=j;

fra=k;

for(j=0,i=x-1; j<x; j++,i--)

{

rem = rem +(a[j] \* pow(8,i));

}

for(k=0,i=1;k<fra;k++,i++)

{

rem1 = rem1 +(b[k] / pow(8,i));

}

rem1 = rem + rem1;

dno = rem1;

num1 = (int)dno;

num2 = dno - num1;

i=0;

while(num1!=0)

{

rem = num1 % 16;

rev[i] = rem;

num1 = num1 /16;

i++;

}

j=0;

while(num2!=0.0)

{

num2 = num2 \* 16;

num3 = (int)num2;

num4 = num2 - num3;

num2 = num4;

a[j] = num3;

j++;

if(j==4)

{

break;

}

}

l=i;

cout<<"\n\tHexadecimal of given number is: ";

for(i=l-1; i>=0; i--)

{

if(rev[i]==10)

{

cout<<"A";

}

else if(rev[i]==11)

{

cout<<"B";

}

else if(rev[i]==12)

{

cout<<"C";

}

else if(rev[i]==13)

{

cout<<"D";

}

else if(rev[i]==14)

{

cout<<"E";

}

else if(rev[i]==15)

{

cout<<"F";

}

else

{

cout<<rev[i];

}

}

h=j;

cout<<".";

for(k=0; k<h; k++)

{

if(a[k]==10)

{

cout<<"A";

}

else if(a[k]==11)

{

cout<<"B";

}

else if(a[k]==12)

{

cout<<"C";

}

else if(a[k]==13)

{

cout<<"D";

}

else if(a[k]==14)

{

cout<<"E";

}

else if(a[k]==15)

{

cout<<"F";

}

else

{

cout<<a[k];

}

}

}

int convertor::HDtoD()

{

char choice;

do{

char hexd[120];

int validn\_count=0,length,lcopy,p=1,num,i,countp=0;

double decimal=0;

do{

cout<<"\n\tEnter a hexadecimal number: ";

cin>>hexd;

length=strlen(hexd);

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

{

if( (hexd[i]>=48 and hexd[i]<=57) or (hexd[i]>=65 and hexd[i]<=70) or (hexd[i]>=97 and hexd[i]<=102) or hexd[i]==46)

++validn\_count;

else

{

validn\_count=0;

cout<<"\n ! WARNING! Invalid HEXADECIMAL NUMBER!!!!";

break;

}

}

}while(validn\_count==0);

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

{

if(hexd[i]==46)

break;

else

++countp;

}

lcopy=countp;

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

{

if(hexd[i]>=48 and hexd[i]<=57)

num=((int)hexd[i]-48);

else if(hexd[i]>=65 and hexd[i]<=70)

num=((int)hexd[i]-55);

else if (hexd[i]>=97 and hexd[i]<=102)

num=((int)hexd[i]-87);

decimal=decimal+(num\*pow(16,--lcopy));

}

for(i=(countp+1);i<length;i++)

{

if(hexd[i]>=48 and hexd[i]<=57)

num=((int)hexd[i]-48);

else if(hexd[i]>=65 and hexd[i]<=70)

num=((int)hexd[i]-55);

else if (hexd[i]>=97 and hexd[i]<=102)

num=((int)hexd[i]-87);

decimal=decimal+(num\*pow(16,-p));

++p;

}

cout<<"\n\tHexadecimal of given number is: "<<decimal;

}while(0);

}

int convertor::HDtoB()

{

char binaryNumber[100],hexaDecimal[100];

long int i=0;

cout<<"\n\tEnter any hexadecimal number: ";

cin>>hexaDecimal;

cout<<"\n\tBinary of given number is: ";

while(hexaDecimal[i])

{

switch(hexaDecimal[i])

{

case '0': cout<<"0000"; break;

case '1': cout<<"0001"; break;

case '2': cout<<"0010"; break;

case '3': cout<<"0011"; break;

case '4': cout<<"0100"; break;

case '5': cout<<"0101"; break;

case '6': cout<<"0110"; break;

case '7': cout<<"0111"; break;

case '8': cout<<"1000"; break;

case '9': cout<<"1001"; break;

case 'A': cout<<"1010"; break;

case 'B': cout<<"1011"; break;

case 'C': cout<<"1100"; break;

case 'D': cout<<"1101"; break;

case 'E': cout<<"1110"; break;

case 'F': cout<<"1111"; break;

case 'a': cout<<"1010"; break;

case 'b': cout<<"1011"; break;

case 'c': cout<<"1100"; break;

case 'd': cout<<"1101"; break;

case 'e': cout<<"1110"; break;

case 'f': cout<<"1111"; break;

default: cout<<"\n\tInvalid hexadecimal digit "<<hexaDecimal[i];

}

i++;

}

}

char convertor::HDtoO()

{

int i,len, dec=0, oct=0;

char hex[20],c;

int n;

cout<<"\n\n\tEnter a hexadecimal number: ";

cin>>hex;

for(len=0; hex[len]!='\0'; len++);

for(i=0; hex[i]!='\0'; i++,len--)

{

if(hex[i]>='0' && hex[i]<='9')

{

dec= dec + (hex[i]-'0')\*pow(16,len-1);

}

if(hex[i]>='A' && hex[i]<='F')

{

dec = dec + (hex[i]-55)\*pow(16,len-1);

}

if(hex[i]>='a' && hex[i]<='f')

{

dec = dec + (hex[i]-87)\*pow(16,len-1);

}

}

i=1;

while(dec!=0)

{

oct = oct + (dec%8)\*i;

dec = dec/8;

i = i\*10;

}

cout<<"\n\n\t Octal of given number is: "<<oct;

}

int main()

{

Arith obj;

Trigo obj1;

Triangle objS;

Rectangle objR;

Trapezoid ObjT;

Ellipse objE;

Square objSQ;

Parallelogram objP;

Circle objC;

Sector objSE;

Cube objCU;

Sphere objSP;

Cone objCON;

Torus objTO;

Iogarithmicfunction obj2;

Powerfunction obj3;

matrix my;

factorial ob;

Factorial ob1;

s\_digit obj4;

Average av;

Percentage perc1;

Prime pric1;

Series obj6;

convertor obj15;

int ch,ch1,ncr,npr;

char ch2;

float a,s,m,d,e,Tri,Tor,Con,Sph,Cub,Sec,Cir,Paral,Squa,Elli,Tra,Rec;;

complex c1,c2,c3;

int choice,cont;

int input;

while(1)

{

cout<<"\n\t\t>>Overloding Overview Calculator<<\n";

cout<<"\n\t 1 : Arithmetic Operations \n";

cout<<"\t 2 : Trigonometric Functions \n";

cout<<"\t 3 : Logarithmic Functions \n";

cout<<"\t 4 : Power Functions \n";

cout<<"\t 5 : complex number\n";

cout<<"\t 6 : matrix \n";

cout<<"\t 7 : add digits\n";

cout<<"\t 8 : n!\n";

cout<<"\t 9 : ncr & npr\n";

cout<<"\t 10 : area calculate\n";

cout<<"\t 11 : summation different type of series\n";

cout<<"\t 12 : average the numbers\n";

cout<<"\t 13 : percentage the number\n";

cout<<"\t 14 : check the prime number\n";

cout<<"\t 15 : NUMBER SYSTERM CONVERTION\n";

cout<<"\t 16 : Exit... \n\n";

cout<<"\n\tEnter the choice: ";

cin>>ch1;

switch(ch1)

{

case 1:

while(1)

{

cout<<"\n\t1: Addition"<<endl;

cout<<"\t2: Subtraction"<<endl;

cout<<"\t3: Multiply"<<endl;

cout<<"\t4: Division"<<endl;

cout<<"\t5: Back"<<endl;

cout<<"\t6: exit"<<endl;

cout<<"\n\tEnter your Choice: ";

cin>>ch;

cout<<endl;

switch(ch)

{

case 1:

a=obj.add();

cout<<"\n\tAddtion Result= "<<a<<endl;

break;

case 2:

s=obj.sub();

cout<<"\n\tSubtract Result= "<<s<<endl;

break;

case 3:

m=obj.mul();

cout<<"\n\tMultiply Result= "<<m<<endl;

break;

case 4:

d=obj.div();

cout<<"\n\tDivision Result= "<<d<<endl;

break;

case 5:

{

main();

break;

}

case 6:

{

exit(0);

break;

}

default:

cout<<"\n\tInvalid choice"<<endl;

}

}

break;

case 2:

while(1)

{

cout<<"\n\t1: sin"<<endl;

cout<<"\t2: cos"<<endl;

cout<<"\t3: tan"<<endl;

cout<<"\t4: sin^-1"<<endl;

cout<<"\t5: cos^-1"<<endl;

cout<<"\t6: tan^-1"<<endl;

cout<<"\t7: Back"<<endl;

cout<<"\t8: exit"<<endl;

cout<<"\n\tEnter your Choice: ";

cin>>ch;

cout<<endl;

switch(ch)

{

case 1:

obj1.Sin();

break;

case 2:

obj1.Cos();

break;

case 3:

obj1.Tan();

break;

case 4:

obj1.Asin();

break;

case 5:

obj1.Acos();

break;

case 6:

obj1.Atan();

break;

case 7:

{

main();

break;

}

case 8:

{

exit(0);

break;

}

default:

cout<<"\n\tInvalid choice"<<endl;

}

}

break;

case 3:

while(1)

{

cout<<"\n\t1: ln"<<endl;

cout<<"\t2: log10"<<endl;

cout<<"\t3: Back"<<endl;

cout<<"\t4: exit"<<endl;

cout<<"\n\tEnter your Choice: ";

cin>>ch;

cout<<endl;

switch(ch)

{

case 1:

obj2.Log();

break;

case 2:

obj2.Log10();

break;

case 3:

{

main();

break;

}

case 4:

{

exit(0);

break;

}

default:

cout<<"\n\tInvalid choice"<<endl;

}

}

break;

case 4:

while(1)

{

cout<<"\t1: pow()"<<endl;

cout<<"\t2: sqrt()"<<endl;

cout<<"\t3: Back"<<endl;

cout<<"\t4: exit"<<endl;

cout<<"\n\tEnter your Choice: ";

cin>>ch;

cout<<endl;

switch(ch)

{

case 1:

obj3.Pow();

break;

case 2:

obj3.Sqrt();

break;

case 3:

{

main();

break;

}

case 4:

{

exit(0);

break;

}

default:

cout<<"\n\tInvalid choice"<<endl;

break;

}

}

break;

case 5:

while(1)

{

cout<<"\n\n\tCOMPLEX NUMBERS";

cout<<"\n\n\t1. ADDITION";

cout<<"\n\n\t2. SUBTRACTION";

cout<<"\n\n\t3. MULTIPLICATION";

cout<<"\n\n\t4. DIVISION";

cout<<"\n\n\t5. Back";

cout<<"\n\n\t6. Exit";

cout<<"\n\n\tEnter your choice : ";

cin>>choice;

if(choice==1||choice==2||choice==3||choice==4)

{

cout<<"\n\n\tEnter the First Complex Number: ";

c1.read();

cout<<"\n\n\tEnter the Second Complex Number: ";

c2.read();

}

switch(choice)

{

case 1 :

c3=c1+c2;

cout<<"\n\n\tSUM = ";

c3.display();

break;

case 2 :

c3=c1-c2;

cout<<"\n\n\tSubtract = ";

c3.display();

break;

case 3 :

c3=c1\*c2;

cout<<"\n\n\tMultiplication = ";

c3.display();

break;

case 4 :

c3=c1/c2;

cout<<"\n\n\tDivision = ";

c3.display();

break;

case 5:

{

cout<<"\n\n\tBack";

main();

break;

}

case 6:

exit(0);

break;

default:

cout<<"\n\n\tUndefined Choice";

break;

}

}

break;

case 6:

while(1)

{

cout<<" \n\t\tMatrix:\n";

cout<<" \t1. Sum of 1st and 2nd matrix\n" ;

cout<<" \t2. Difference of 1st and 2nd matrix\n";

cout<<" \t3. Transpose of both 1st and 2nd matrix\n";

cout<<" \t4. multiflication of 1st and 2nd matrix\n";

cout<<" \t5. Back\n";

cout<<" \t6. Exit\n";

cout<<" \n\t Enter your choice: ";

cin>> input;

switch (input)

{

case 1:

my.sum2();

break;

case 2:

my.diff();

break;

case 3:

my.transpose();

break;

case 4:

my.multifica1();

break;

case 5:

main();

break;

case 6:

exit(0);

break;

}

}

cout<< "\n";

return 0;

break;

case 7:

while(1)

{

cout<<"\n\n\t1. Add digits";

cout<<"\n\t2. BACK";

cout<<"\n\t3. EXIT";

cout<<"\n\tEnter the choice: ";

cin>>ch1;

cout<<endl;

switch(ch1)

{

case 1:

obj4.input();

obj4.calculate();

obj4.display();

break;

case 2:

main();

break;

case 3:

exit(0);

break;

default:

cout<<"\n\tInvalid choice\n";

}

}

getch();

break;

case 8:

while(1)

{

cout<<"\n\t1. n!";

cout<<"\n\t2. BACK";

cout<<"\n\t3. EXIT";

cout<<"\n\tEnter the choice: ";

cin>>ch1;

cout<<endl;

switch(ch1)

{

case 1:

ob.fact();

ob.display();

break;

case 2:

main();

break;

case 3:

exit(0);

break;

default:

cout<<"\n\tInvalid choice\n";

}

}

getch();

break;

case 9:

while(1)

{

cout<<"\n\t1. nCr";

cout<<"\n\t2. nPr";

cout<<"\n\t3. BACK";

cout<<"\n\t4. EXIT";

cout<<"\n\tEnter the choice: ";

cin>>ch1;

cout<<endl;

switch(ch1)

{

case 1:

ob1.Factn();

ob1.Factr();

ob1.Factnr();

ob1.display();

break;

case 2:

ob1.Factn();

ob1.Factr();

ob1.Factnr();

ob1.display1();

break;

case 3:

main();

break;

case 4:

exit(0);

break;

default:

cout<<"\n\tInvalid choice\n";

}

}

getch();

break;

case 10:

while(1)

{

cout<<"\n\t1: triangle"<<endl;

cout<<"\t2: rectangle"<<endl;

cout<<"\t3: trapezoid"<<endl;

cout<<"\t4: ellipse"<<endl;

cout<<"\t5: square"<<endl;

cout<<"\t6: parallelogram"<<endl;

cout<<"\t7: circle"<<endl;

cout<<"\t8: sector"<<endl;

cout<<"\t9: cube"<<endl;

cout<<"\t10: sphere"<<endl;

cout<<"\t11: cone"<<endl;

cout<<"\t12: torus"<<endl;

cout<<"\t13: Back"<<endl;

cout<<"\t14: exit"<<endl;

cout<<"\n\tEnter your Choice: ";

cin>>ch;

cout<<endl;

switch(ch)

{

case 1:

objS.getNumber();

objS.get1Number();

Tri =objS.getTriangle();

cout << "\n\tTriangle of area = "<< Tri << endl;

break;

case 2:

objR.get7Number();

objR.get8Number();

Rec =objR.getRectangle();

cout << "\n\tRectangle of area = "<< Rec << endl;

break;

case 3:

ObjT.get1Number();

ObjT.get2Number();

ObjT.get3Number();

Tra =ObjT.getTrapezoid();

cout << "\n\tTrapezoid of area = "<< Tra << endl;

break;

case 4:

objE.get2Number();

objE.get3Number();

Elli =objE.getEllipse();

cout << "\n\tEllipse of area = "<< Elli << endl;

break;

case 5:

objSQ.get6Number();

Squa =objSQ.getSquare();

cout << "\n\tSquare of area = "<< Squa << endl;

break;

case 6:

objP.getNumber();

objP.get1Number();

Paral =objP.getParallelogram();

cout << "\n\tParallelogram of area = "<< Paral << endl;

break;

case 7:

objC.get5Number();

Cir =objC.getCircle();

cout << "\n\tCircle of area = "<< Cir << endl;

break;

case 8:

objSE.get5Number();

objSE.get4Number();

Sec =objSE.getSector();

cout << "\n\tSector of area = "<< Sec << endl;

break;

case 9:

objCU.get5Number();

Cub =objCU.getCube();

cout << "\n\tCube of area = "<< Cub << endl;

break;

case 10:

objSP.get5Number();

Sph =objSP.getSphere();

cout << "\n\tSphere of area = "<< Sph << endl;

break;

case 11:

objCON.get5Number();

objCON.get2Number();

Con =objCON.getCone();

cout << "\n\tCone of area = "<< Con << endl;

break;

case 12:

objTO.get2Number();

objTO.get3Number();

Tor =objTO.getTorus();

cout << "\n\tTorus of area = "<< Tor << endl;

break;

case 13:

{

main();

break;

}

case 14:

{

exit(0);

break;

}

default:

cout<<"\n\tInvalid choice"<<endl;

}

}

break;

case 11:

while(1)

{

cout<<"\n\t1: Harmonic Progression(H.P)series"<<endl;

cout<<"\t2: Geometric Progression(G.P)series"<<endl;

cout<<"\t3: Arithmetic Progression(A.P)series"<<endl;

cout<<"\t4: Back"<<endl;

cout<<"\t5: exit"<<endl;

cout<<"\n\tEnter your Choice: ";

cin>>ch;

cout<<endl;

switch(ch)

{

case 1:

obj6.HPseries();

break;

case 2:

obj6.GPseries();

break;

case 3:

obj6.APseries();

break;

case 4:

{

main();

break;

}

case 5:

{

exit(0);

break;

}

default:

cout<<"\n\tInvalid choice"<<endl;

}

}

break;

case 12:

while(1)

{

cout<<"\n\t1. Average the number";

cout<<"\n\t2. BACK";

cout<<"\n\t3. EXIT";

cout<<"\n\tEnter the choice: ";

cin>>ch1;

cout<<endl;

switch(ch1)

{

case 1:

av.add1();

break;

case 2:

main();

break;

case 3:

exit(0);

break;

default:

cout<<"\n\tInvalid choice\n";

}

}

getch();

break;

case 13:

while(1)

{

cout<<"\n\t1. Percentage";

cout<<"\n\t2. BACK";

cout<<"\n\t3. EXIT";

cout<<"\n\tEnter the choice: ";

cin>>ch1;

cout<<endl;

switch(ch1)

{

case 1:

perc1.per();

break;

case 2:

main();

break;

case 3:

exit(0);

break;

default:

cout<<"\n\tInvalid choice\n";

}

}

getch();

break;

case 14:

while(1)

{

cout<<"\n\n\t1. Check Prime Number";

cout<<"\n\t2. BACK";

cout<<"\n\t3. EXIT";

cout<<"\n\tEnter the choice: ";

cin>>ch1;

cout<<endl;

switch(ch1)

{

case 1:

pric1.prime();

break;

case 2:

main();

break;

case 3:

exit(0);

break;

default:

cout<<"\n\tInvalid choice\n";

}

}

getch();

break;

case 15:

while(1)

{

cout<<"\n\n\t 1: Decimal TO Binery"<<endl;

cout<<"\t 2: Decimal TO Octal"<<endl;

cout<<"\t 3: Decimal TO HexaDecimal"<<endl;

cout<<"\t 4: Binery TO Decimal"<<endl;

cout<<"\t 5: Binery TO Octal"<<endl;

cout<<"\t 6: Binery TO HexaDecimal"<<endl;

cout<<"\t 7: Octal TO Decimal"<<endl;

cout<<"\t 8: Octal TO Binery"<<endl;

cout<<"\t 9: Octal TO HexaDecimal"<<endl;

cout<<"\t10: HexaDecimal TO Decimal"<<endl;

cout<<"\t11: HexaDecimal TO Binery"<<endl;

cout<<"\t12: HexaDecimal TO Octal"<<endl;

cout<<"\t13: Back"<<endl;

cout<<"\t14: exit"<<endl;

cout<<"\n\tEnter your Choice: ";

cin>>ch;

cout<<endl;

switch(ch)

{

case 1:

obj15.DtoB();

break;

case 2:

obj15.DtoO();

break;

case 3:

obj15.DtoHD();

break;

case 4:

obj15.BtoD();

break;

case 5:

obj15.BtoO();

break;

case 6:

obj15.BtoHD();

break;

case 7:

obj15. OtoD();

break;

case 8:

obj15. OtoB();

break;

case 9:

obj15. OtoHD();

break;

case 10:

obj15. HDtoD();

break;

case 11:

obj15. HDtoB();

break;

case 12:

obj15. HDtoO();

break;

case 13:

{

main();

break;

}

case 14:

{

exit(0);

break;

}

default:

cout<<"\n\tInvalid choice"<<endl;

break;

}

}

break;

case 16:

{

exit(0);

break;

}

default:

cout<<"\n\tInvalid choice"<<endl;

break;

}

}

}

**Future plan about our project:**

1.Remove all limitation in our project.  
2.Extantion all functionaries activities.  
3.Make perfect user useable calculator such as we use scientific calculator.