# PROJECT REPORT

ON

# *Scientific Calculator*


SUBMITTED BY :

NAME - D. Murali krishna

BATCH - ITAP 203

CRANES VARSITY

BENGALURU

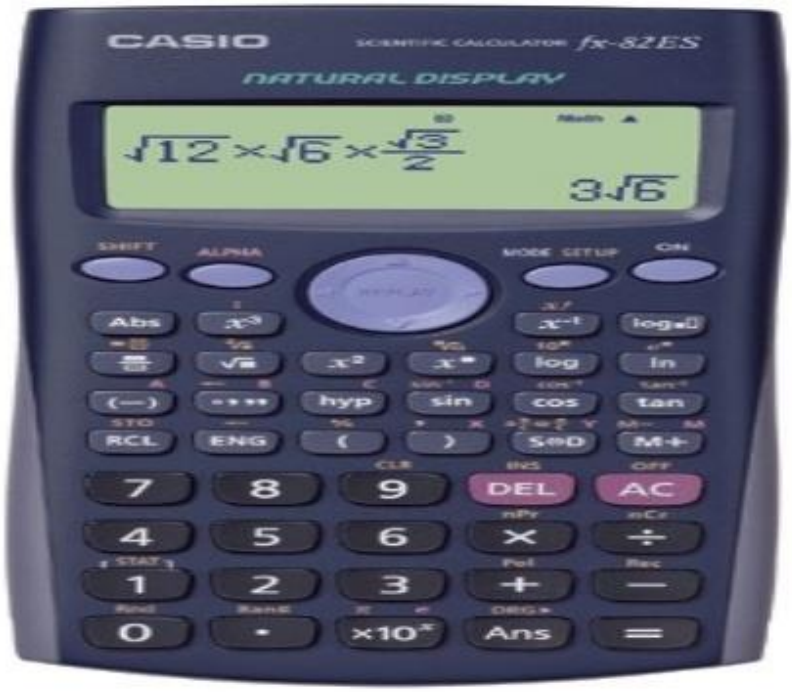
GUIDED BY :

ATMAJA MADAM

CRANES VARSITY

BENGALURU

# *SCIENTIFIC CALCULATOR*



# TABLE OF CONTENTS

# INTRODUCTION

## Scientific Calculator :

The calculator was written by Rolf Howarth in early 1996.

A fully featured scientific calculator with proper operator precedence is implemented, including trig functions and logarithms, factorials, 12 levels of parentheses, logs to base 2 (a handy function for information entropists!), bitwise logical operators, hex, octal, binary and ASCII display.

The calculator is written in JavaScript and you are welcome to view the JavaScript source (visible within the HTML page) for personal educational purposes as long as you recognize that it is copyrighted and *not* in the public domain. This calculator is now available as part of Hummingbird's Enterprise Information Portal. All enquiries regarding licensing the calculator should be directed to Hummingbird Ltd.

# BASIC FUNCTIONS

### Addition

The addition (sum function) is used by clicking on the "+" button or using the keyboard. The function results in a+b.

### Subtraction

The subtraction (minus function) is used by clicking on the "-" button or using the keyboard.

The function results in a-b.

### Multiplication

The multiplication (times function) is used by clicking on the "x" button or using the keyboard "*" key. The function results in a*b.

### Division

The division (divide function) is used by clicking on the "/" button or using the keyboard "/" key. The function results in a/b.

### Sign

The sign key (negative key) is used by clicking on the "(-)" button. The function results in -1*x.

### Square

The square function is used by clicking on the "x^2" button or type "^2". The function results in x*x.

### Square Root

The square root function is used by clicking on the "x" button or type "sqrt()". This function represents x^.5 where the result squared is equal to x.

### Raise to the Power

The raise to the power (y raised to the x function) is used by clicking on the "y^x" button or type "^".

### Natural Exponential

The natural exponential (e raised to the x) is used by clicking on the "e^x" button or type "exp()". The result is e (2.71828...) raised to x.

### Logarithm

The logarithm (LOG) is used by clicking on the "LOG" button or type "LOG()".

### Natural Logarithm

The Natural logarithm (LN) is used by clicking on the "LN" button or type "LN()".

### Inverse

Multiplicative inverse (reciprocal function) is used by pressing the "1/x" button or typing "inv()". This function is the same as x^-1 or dividing 1 by the number.

### Exponent

Numbers with exponents of 10 are displayed with an "e", for example 4.5e+100 or 4.5e-100. This function represents 10^x. Numbers are automatically displayed in the format when the number is too large or too small for the display. To enter a number in this format use the exponent key "EEX". To do this enter the mantissa (the non exponent part) then press "EEX" or type"e" and then enter the exponent.

### Factorial

The Factorial function is used by clicking the "!" button or type "!".

### PI

PI is a mathematical constant of the ratio of a circle's circumference to its diameter.

# PROPOSED SYSTEM

The following documentation is a project the “Name of the term paper allotted”. It is a detailed summary of all the drawbacks of the old system and how the new proposed system overcomes these shortcomings. The new system takes into account the various factors while designing a new system. It keeps into the account the Economical bandwidth available for the new system. The foremost thing that is taken care of is the Need and Requirements of the User.

**DESCRIPTION**

Before developing software we keep following things in mind that we can develop powerful and quality software

PROBLEM STATEMENT

- Problem statement was to design a module:
- Which is user friendly
- Which will restrict the user from accessing other user’s data.
- Which will help user in viewing his data and privileges.
- Which will help the administrator to handle all the changes.

FUNCTIONS TO BE PROVIDED:

The system will be user friendly and completely menu driven so that the users shall have no problem in using all options.

- The system will be efficient and fast in response.
- The system will be customized according to needs.

**SYSTEM REQUIRMENTS**

Operating system: MS Windows XP or Windows 10

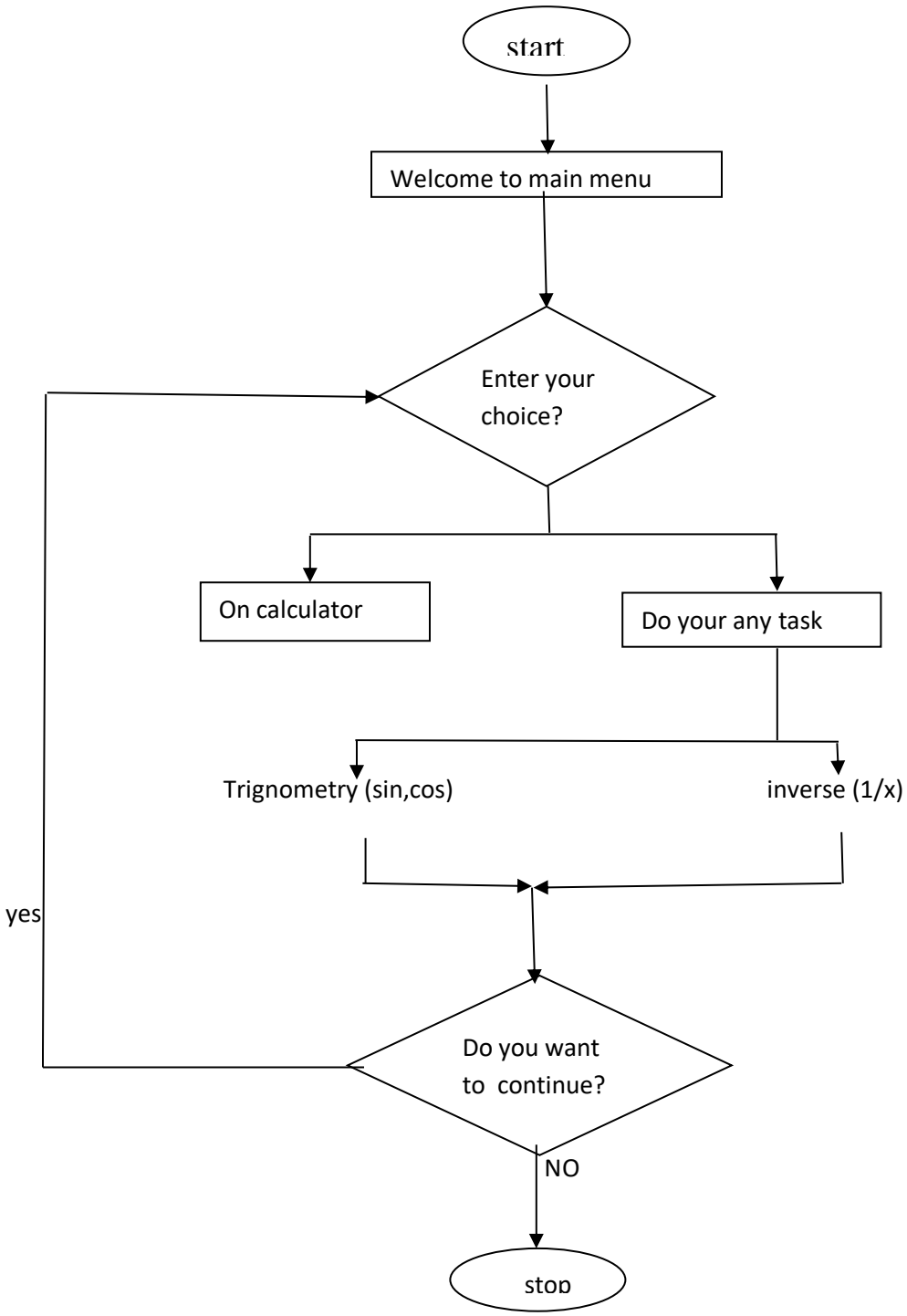
Language: C Language

Processor: Pentium IV Processor RAM: 512 MB Hard disk: 2 GB

# SYSTEM DESIGN

Then we began with the design phase of the system. System design is a solution, a “HOW TO” approach to the creation of a new system. It translates system requirements into ways by which they can be made operational. It is a translational from a user oriented document to a document oriented programmers. For that, it provides the understanding and procedural details necessary for the implementation. Here we use Flowchart to supplement the working of the new system. The system thus made should be reliable, durable and above all should have least possible maintenance costs. It should overcome all the drawbacks of the Old existing system and most important of all meet the user requirements.

# FLOW CHART


start
Welcome to main menu
Enter your choice?
On calculator
Do your any task
Trignometry (sin,cos)
inverse (1/x)
yes
Do you want to continue?
NO
stop

# CODING:

```
#include <stdio.h>
#include<stdlib.h>
#include<math.h>
int addition();
int subtraction();
int multiplication();
int division();
int modulus();
int factorial();
int power();
int square();
int squareroot();
int absolute();
int naturallogarithm();
int exponentiallogarithm();
int decimaltobinary();
int binarytodecimal();
int sine();
int cosine();
int tann();
int cosec();
int sec();
int cot();
int main()
{
int option;
printf("enter your option\n");
scanf("%d",&option);
switch(option)
{
case 1: addition();
break;
case 2: subtraction();
break;
case 3: multiplication();
break;
case 4: division();
break;
case 5: modulus();

break;
case 6: factorial();
break;
case 7: power();
```

```
break;
case 8: square();
break;
case 9: squareroot();
break;
case 10: absolute();
break;
case 11: naturallogarithm();

break;
case 12: exponentiallogarithm();
break;
case 13: sine();
break;
case 14: cosine();
break;
case 15: tann();
break;
case 16: cosec();
break;
case 17: sec();
break;
case 18: cot();
break;
case 19: decimaltobinary();
break;
case 20: binarytodecimal();
break;
default : return 0;
break;
}
}
int addition()
{
int a,b,c;
printf("performing addition operation\n");
printf("enter a and b\n");
scanf("%d%d",&a,&b);
c=a+b;
printf("addition = %d\n",c);
}
int subtraction()
{
int a,b,c;
printf("performing subtraction operation\n");
printf("enter a and b\n");
```

```
scanf("%d%d",&a,&b);
c=a-b;
printf("subtraction = %d\n",c);
}
int multiplication()
{
int a,b,c;
printf("performing multiplication operation\n");
printf("enter a and b\n");
scanf("%d%d",&a,&b);
c=a*b;
printf("multiplication = %d\n",c);
}
int division()
{
int a,b,c;
printf("performing division operation\n");
printf("enter a and b\n");
scanf("%d%d",&a,&b);
c=a/b;
printf("division %d\n",c);
}
int modulus()
{
int a,b,c;
printf("performing modulus operation\n");
printf("enter a and b\n");
scanf("%d%d",&a,&b);
c=a%b;
printf("modulus %d\n",c);
}
int factorial()
{
int n,fact=1,i;
printf("performing factorial operation\n");
printf("enter n\n");
scanf("%d",&n);
for(i=1;i<=n;i++)
{
fact=fact*i;
}
printf("factorial %d\n",fact);
}
int power()
{
int base,exponential,result;
printf("performing power operation\n");
printf("enter base and exponential values\n");
```

```
scanf("%d%d",&base,&exponential);
result=pow(base,exponential);
printf("result = %d\n",result);
}
int square()
{
int a,result;
printf("perform square operation\n");
printf("enter a\n");
scanf("%d",&a);
result=a*a;
printf("result = %d\n",result);
}
int squareroot()
{
int a,result;
printf("performing squareroot operation\n");
printf("enter a\n");
scanf("%d",&a);
result=sqrt(a);
printf("result = %d\n",result);
}
int absolute()
{
int a,result;
printf("perform absolute operation\n");
printf("enter a\n");
scanf("%d",&a);
result=abs(a);
printf("result %d\n",result);
}
int naturallogarithm()
{
double a,result;
printf("performing natural logarithm operation\n");
printf("enter a\n");
scanf("%lf",&a);
result=log10(a);
printf("result = %lf",result);
}
int exponentiallogarithm()
{
double a,result;
printf("performing exponential logarithm operation\n");
printf("enter a\n");
scanf("%lf",&a);
result=log(a);
printf("result = %lf",result);
```

```
}
int sine()
{
float x,result;
printf("perform sine operation\n");
printf("enter x\n");
scanf("%f",&x); //30
result=sin(x*3.14/180);
printf("%.2f",result);
}
int cosine()
{
float x,result;
printf("perform cosine operation\n");
printf("enter x\n");
scanf("%f",&x);
result=cos(x*3.14/180);
printf("%.2f",result);
}
int tann()
{
float x,result;
printf("perform tann operation\n");
printf("enter x\n");
scanf("%f",&x);
result=tan(x*3.14/180);
printf("%.2f",result);
}
int cosec()
{
float x,result;
printf("perform cosec operation\n");
printf("enter x\n");
scanf("%f",&x);
result=(1/sin(x*3.14/180));
printf("%.2f",result);
}
int sec()
{
float x,result;
printf("perform sec operation\n");
printf("enter x\n");
scanf("%f",&x); //30
result=(1/cos(x*3.14/180));
printf("%.2f",result);
}
int cot()
```

```
{
float x,result;
printf("perform cot operation\n");
printf("enter x\n");
scanf("%f",&x); //30
result=(1/tan(x*3.14/180));
printf("%.2f",result);
}
int decimaltobinary()
{
int b[100],n,i=0,j;
printf("performing decimal to binary operation\n");
printf("enter n\n");
scanf("%d",&n);
while(n>0)
{
b[i]=n%2;
n=n/2;
i++;
}
for(j=i-1;j>=0;j--)
{
printf("%d",b[j]);
}
}
int binarytodecimal()
{
int n,i,r,sum=0;
printf("performing binary to decimal operation\n");
printf("enter n\n");
scanf("%d",&n);
i=0;
while(n>0)
{
r=n%10;
sum=sum+r*pow(2,i);
n=n/10;
i++;
}
printf("%d",sum);
}
```

# APPLICATIONS

In most countries, students use calculators for schoolwork. There was some initial resistance to the idea out of fear that basic arithmetic skills would suffer. There remains disagreement about the importance of the ability to perform calculations "in the head", with some curricula restricting calculator use until a certain level of proficiency has been obtained, while others concentrate more on teaching estimation techniques and problem-solving. Research suggests that inadequate guidance in the use of calculating tools can restrict the kind of mathematical thinking that students engage in. Others have argued that calculator use can even cause core mathematical skills to atrophy, or that such use can prevent understanding of advanced algebraic concepts.

There are other concerns - for example, that a pupil could use the calculator in the wrong fashion but believe the answer because that was the result given. Teachers try to combat this by encouraging the student to make an estimate of the result manually and ensuring it roughly agrees with the calculated result. Also, it is possible for a child to type in $-1 \times -1$ and obtain the correct answer '1' without realizing the principle involved. In this sense, the calculator becomes a crutch rather than a learning tool, and it can slow down students in exam conditions as they check even the most trivial result on a calculator.

# FUTURE SCOPE OF THE PROJECT

Our project will be able to implement in future after making some changes and modifications as we make our project at a very low level. So the modifications that can be done in our project are:

To make it screen touch so no need to touch key buttons and one more change which can we made is to add snaps of the person who use it.

# TESTING

Testing is the major control measure used during software development. Its basic function is to detect errors in the software. During requirement analysis and design, the output is a document that is usually textual and no executable. After the coding phase, computer programs are available that can be executed for testing purpose. This implies that testing not only, has to uncover errors introduced during coding, but also errors introduced during previous phase. Thus the goal of testing is to uncover the requirements, design and coding errors in the programs. TheSourcecode declared above for the program of Scientific Calculator has been tested and it has been found that the above source code is okay and correct.The program involves many type of conversions. These conversions has to done carefully.

# REFERENCES


- Thomas J. Bing, Edward F. Redish, Symbolic Manipulators Affect Mathematical Mindsets, December 2007

**1.** ^ Mike Sebastian's calculator forensics algorithm is an example of such rounding errors -- the algorithm's *arcsin(arccos(arctan(tan(cos(sin(9))))))* should come out 9 on standard floating point hardware, but for CORDIC it's a pathological case that produces different rounding errors on each chip that it is implemented on. The algorithm is primarily used to identify the manufacturer of a particular calculator's CPU, since it is usually reproducible between chips of the same model.

2. ^ Georges Ifrah notes that humans learned to count on their hands. Ifrah shows, for example, a picture of Boethius (who lived 480–524 or 525) reckoning on his fingers in Ifrah 2000, p. 48.

3. ^ According to Schmandt-Besserat 1981, these clay containers contained tokens, the total of which were the count of objects being transferred. The containers thus served as a bill of lading or an accounts book. In order to avoid breaking open the containers, marks were placed on the outside of the containers, for the count. Eventually (Schmandt-Besserat estimates it took 4000 years) the marks on the outside of the containers were all that were needed to convey the count, and the clay containers evolved into clay tablets with marks for the count.

**4.** ^ Lazos 1994

5. ^ *Ancient Discoveries, Episode 11: Ancient Robots*, History Channel, http://www.youtube.com/watch?v=rxjbaQl0ad8, retrieved on 6 September 2008

6. ^ A Spanish implementation of Napier's bones (1617), is documented in Montaner i Simon 1887, pp. 19-20.

**7.** ^ Kells, Kern & Bland 1943, p. 92

**8.** ^ Kells, Kern & Bland 1943, p. 82, as log(2)=.3010, or 4 places.

**9.** ^ Schmidhuber

**10.** ^ As quoted in Smith 1929, pp. 180-181