Experiment no. 2

Introduction to C programming language

1.Introduction

C is a general purpose programming language. It is closely associated with Unix system where it was developed, since both the system and most of the programs that run on it are written in C.

C is a relatively low level language and is used most when we need to closely interact with hardware like in Operating Systems or Device Driver code.

C programming language was developed in 1972 by Dennis Ritchie at Bell Laboratories of AT&T

2. Environment setup

Following 2 software tools would be used for setting up the environment for C programming

1)Text Editor - We would be using gedit for this course

2)C compiler - We would be using gcc for this course

Open terminal and write following at the shell prompt to open up a Text editor using gedit

```
$gedit <filename.c>
```

C program files must end with .c extension

Check for gcc compiler using following command

```
$gcc -v
```

In case there is no gcc compiler, download it using following command

```
$sudo apt-get update
$sudo apt-get install gcc
```

Example 1

First C program Hello World

1.Create a directory to store your C programs. Go inside that directory

```
$mkdir C_Programs
$cd C_Programs
```

2.Open text editor to create a file helloworld.c

```
$gedit helloworld.c
```

3.Type following C program into helloworld.c

```
#include <stdio.h>

int main() {
   /* my first program in C */
   printf("Hello, World! \n");

   return 0;
}
```

4 Save helloworld.c using Ctrl + S

5 Close the Text file helloworld.c either by pressing x button on GUI or by pressing Ctrl + C on Terminal

6 Compile helloworld.c using gcc

```
$gcc helloworld.c
```

7 Execute the newly generated executable file a.out in the same directory.

```
$./a.out
```

Some Points to Note about the C program

1. #include <stdio.h> includes the standard input output library functions. The printf() function is defined in stdio.h file.
2. int main() - The main() function is the entry point of every program in c language.
3. printf() - The printf() function is used to print data on the console.
4. return 0 - The return 0 statement, returns execution status to the OS. The 0 value is used for successful execution of the program

3. `printf()` and `scanf()` functions in C

1) `printf()` function

This function is used to print to the console

Syntax

```
printf("format string",argument_list);
```

The format string can be %d for integer, %c for character, %s for string, %f for float etc.

2) `scanf()` function

This function is used for input. It reads the input data from the console.

Syntax

```
scanf("format string",argument_list);
```

Here also the format string can be %d for integer, %c for character, %s for string, %f for float etc.

Argument list usually have address of variables that store data

Example 2

Program to find sum of 2 numbers

```
#include<stdio.h>
int main(){
      int x=0,y=0,result=0;
      printf("enter first number:");
      scanf("%d",&x);
      printf("enter second number:");
      scanf("%d",&y);
      result=x+y;
      printf("sum of 2 numbers:%d ",result);
return 0;
}
```

4.Comments in C

Comments are added with the purpose of making the source code easier for humans to understand, and are generally ignored by compilers and interpreters

There are 2 types of comments in C language.

1 Single Line Comments

2 Multiline Comments

Example 3

Example for Single Line Comments

Single line comments are represented by double forward slash //

```
#include<stdio.h>
int main(){
    //This is a single line comment
    printf("Hello C");    // Printing Information
return 0;
}
```

Example 4

Example for Multiline Comment

Multi line comments are represented by slash asterisk /* … */. So everything inside /* */ can occupy many lines of code and would be ignored by C compiler but it can't be nested.

```
#include<stdio.h>

int main(){

    /*printing information
      Multi Line Comment*/

    printf("Hello C");
return 0;
}
```

5.Data Types in C

Each variable in C has an associated data type. Each data type requires different amounts of memory and has some specific operations which can be performed over it.

Type	Data Type
Basic Data Type	int, char, float, long,short,double
Derived Data Type	array, pointer, structure, union
Enumeration Data Type	enum
Void Data Type	void

Data Type	Range	Bytes	Format
signed char	-128 to + 127	1	%c
unsigned char	0 to 255	1	%c
short signed int	-32768 to +32767	2	%d
short unsigned int	0 to 65535	2	%u
signed int	-32768 to +32767	2	%d
unsigned int	0 to 65535	2	%u
long signed int	-2147483648 to +2147483647	4	%ld
long unsigned int	0 to 4294967295	4	%lu
float	-3.4e38 to +3.4e38	4	%f
double	-1.7e308 to +1.7e308	8	%lf
long double	-1.7e4932 to +1.7e4932	10	%Lf

Note: The sizes and ranges of int, short and long are compiler dependent. Sizes in this figure are for 16-bit compiler.

6. Variables in C

A variable is a name of memory location. It is used to store data.

Syntax to declare a variable

```
variable_type variable_list;
```

Example

```
int a;
float b;
char c;
```

Here, a, b, c are variables and int,float,char are data types.

We can also provide values while declaring the variables as given below:

```
int a=10,b=20;//declaring 2 variable of integer type
float f=20.8;
char c='A';
```

Rules for Defining Variables

1. A variable can have alphabets, digits and underscore.
2. A variable name can start with alphabet and underscore only. It can't start with digit.
3. No white space is allowed within variable name.
4. A variable name must not be any reserved word or keyword e.g. int, float etc.

Example

Valid variable names

```
int a;
int _ab;
int a30;
```

Invalid Variable names

```
int 2;
int a b;
int long;
```

We can use the sizeof() operator to check the size of variable of a particular data type in memory.

Example 5

```
#include<stdio.h>
int main()
{
    int a = 1;
    char b ='G';
    double c = 3.14;
    printf("Hello World!\n");

    //printing the variables defined above along with their sizes
    printf("Hello! I am a character. My value is %c and "
           "my size is %lu byte.\n", b,sizeof(char));

    //can use sizeof(b) above as well

    printf("Hello! I am an integer. My value is %d and "
           "my size is %lu  bytes.\n", a,sizeof(int));

    //can use sizeof(a) above as well
```

```
printf("Hello! I am a double floating point variable."
           " My value is %lf and my size is %lu
bytes.\n",c,sizeof(double));


    //can use sizeof(c) above as well


    printf("Bye! See you soon. :)\n");


    return 0;
}
```

7.Operators in C

An operator is a symbol that tells the compiler to perform specific mathematical or logical functions.

C has following operators in order of their precedence

Description	Operator	Associativity
Function expression	( )	Left to Right
Array Expression	[ ]	Left to Right
Structure operator	->	Left to Right
Structure operator	.	Left to Right
Unary minus	-	Right to left
Increment/Decrement	++ --	Right to Left
One's compliment	~	Right to left
Negation	!	Right to Left
Address of	&	Right to left
Value of address	*	Right to left
Type cast	(type)	Right to left
Size in bytes	sizeof	Right to left
Multiplication	*	Left to right
Division	/	Left to right
Modulus	%	Left to right
Addition	+	Left to right
Subtraction	-	Left to right
Left shift	<<	Left to right
Right shift	>>	Left to right
Less than	<	Left to right
Less than or equal to	<=	Left to right
Greater than	>	Left to right
Greater than or equal to	>=	Left to right
Equal to	==	Left to right
Not equal to	!=	Left to right

8. Escape Sequences in C

An escape sequence in C language is a sequence of characters that doesn't represent itself when used inside string literal or character.

It is composed of two or more characters starting with backslash \. For example: \n represents new line.

Esc. Seq.	Purpose	Esc. Seq.	Purpose
\n	New line	\t	Tab
\b	Backspace	\r	Carriage return
\f	Form feed	\a	Alert
\'	Single quote	\"	Double quote
\\	Backslash		

Example 6

```
#include<stdio.h>
int main(){
  int number=50;
  printf("You\nare\nlearning\n\'c\' language\n\"Do you know C language\"");
 return 0;
}
```

9. Constants in C

A constant is a value or variable that can't be changed in the program. It can be a decimal, floating point, octal, hexadecimal, character or string.

For example 10, 20, 'a', 3.4, "c programming" etc.

Ways to define a constant

1. const keyword
2. #define preprocessor directive

Example 7

```
#include<stdio.h>
int main(){
     const float PI=3.14;
     PI=4.5;
        printf("The value of PI is: %f",PI);
    return 0;
}
```

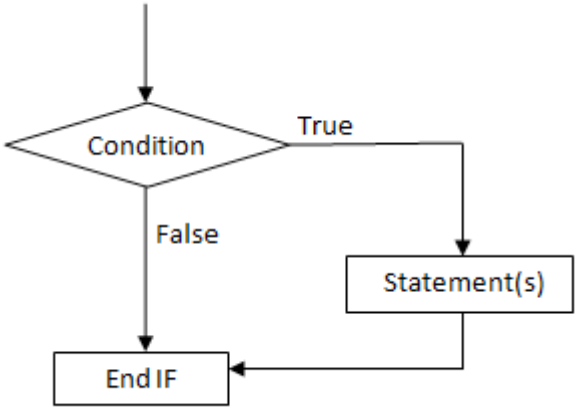
Example 8

```
#include<stdio.h>
#define PI 3.14
int main(){
     PI=4.5;
        printf("The value of PI is: %f",PI);
    return 0;
}
```

Above examples should throw a compile time error as PI once declared a constant can't be modified

10. if else in C

If Statement



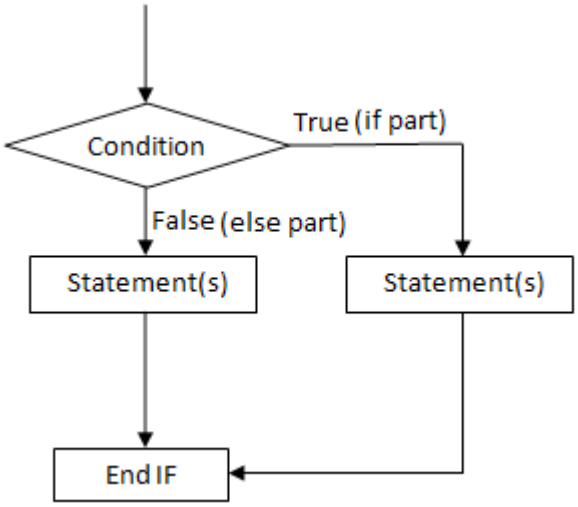
Example 9

C program to print the square of a number if it is less than 10

```
#include<stdio.h>
int main()
{
    int n;
    printf("Enter a number:");
    scanf("%d",&n);
    if(n<10)
    {
        printf("%d is less than 10\n",n);
        printf("Square = %d\n",n*n);
    }
```

```
    return 0;
}
```

If…else Statement



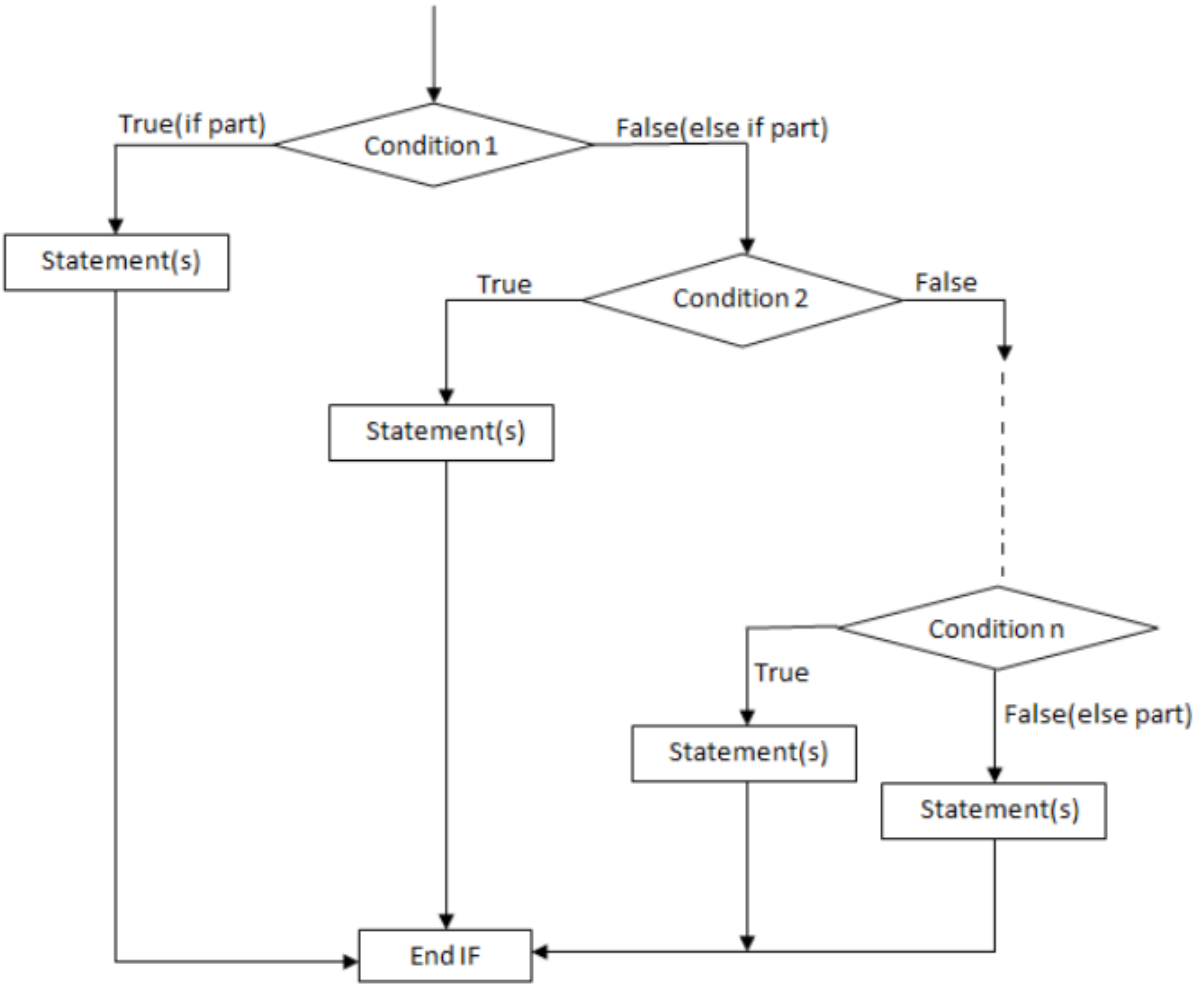
Example 10

C program to find if a number is odd or even.

```
#include<stdio.h>
int main()
{
    int n;
    printf("Enter a number:");
    scanf("%d",&n);
    if(n%2 == 0)
        printf("%d is even",n);
    else
        printf("%d is odd",n);
    return 0;
```

```
}
```

if … else if … else



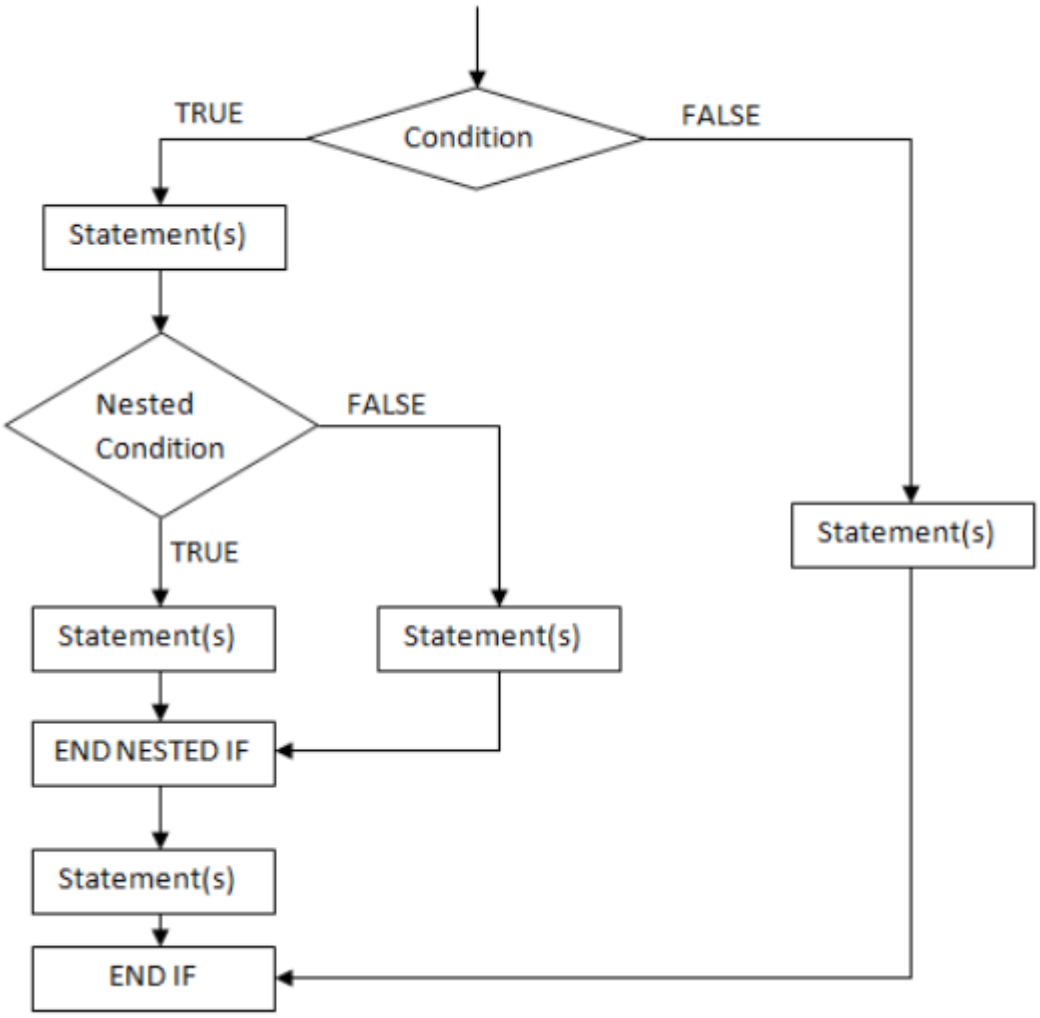
Example 11

C program to find if a number is negative, positive or zero.

```
#include<stdio.h>
int main()
{
```

```
    int n;
    printf("Enter a number:");
    scanf("%d",&n);
    if(n<0)
        printf("Number is negative");
    else if(n>0)
        printf("Number is positive");
    else
        printf("Number is equal to zero");
    return 0;
}
```

Nested if statements



Example 12

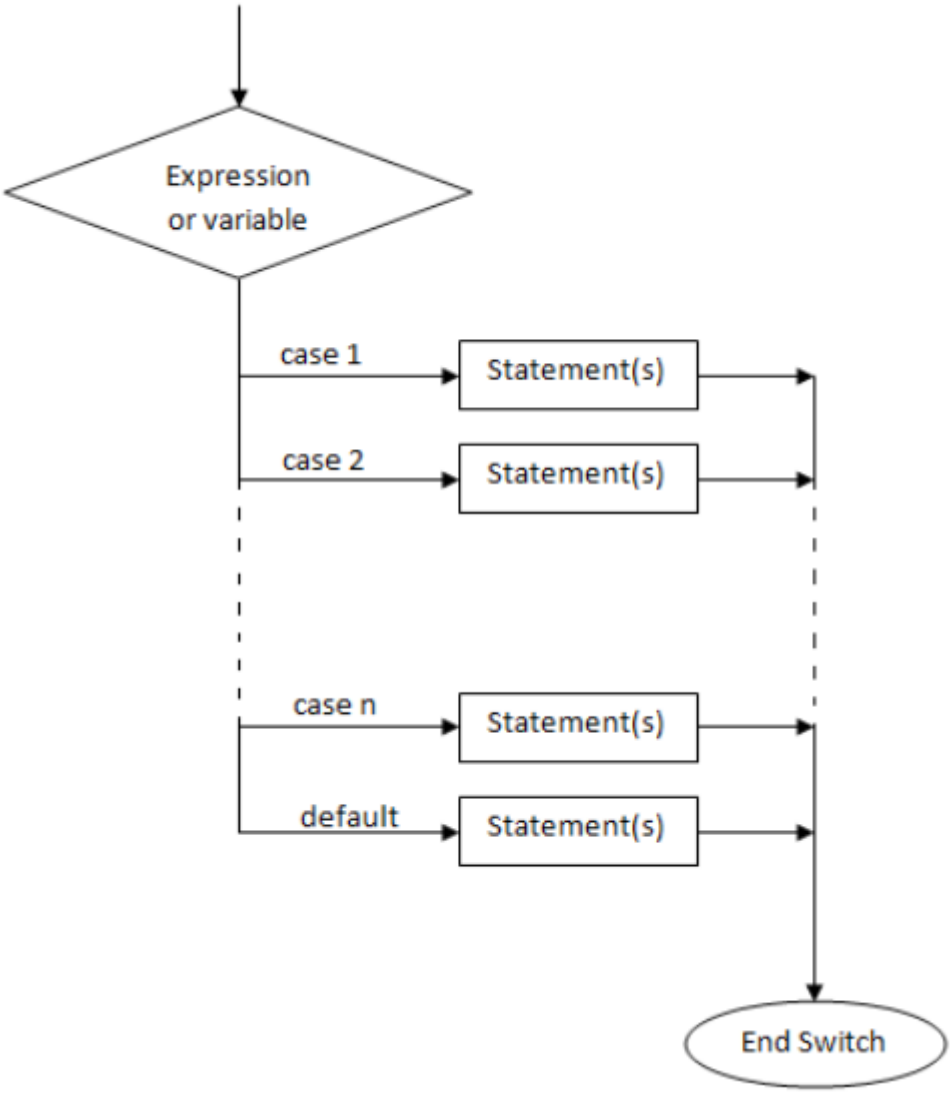
C program to check if a number is less than 100 or not. If it is less than 100 then check if it is odd or even.

```
#include<stdio.h>
int main()
{
    int n;
    printf("Enter a number:");
    scanf("%d",&n);
```

```
    if(n<100)
    {
        printf("%d is less than 100\n",n);
        if(n%2 == 0)
            printf("%d is even",n);
        else
            printf("%d is odd",n);
    }
    else
        printf("%d is equal to or greater than 100",n);
    return 0;
}
```

11. Switch statement in C

Switch case is a multiple branching statement which compares the value of expression or variable inside switch() with various cases provided with the statement and executes a block when a match is found. If no cases inside the switch is matched, the statements inside default block is executed. However, default is optional and may not be present. Default is similar to else part of if statement.



Example 13

Check if entered alphabet is vowel or a consonant

```
#include <stdio.h>
int main()
{
    char alphabet;
    printf("Enter an alphabet:");
    scanf("%c",&alphabet);
```

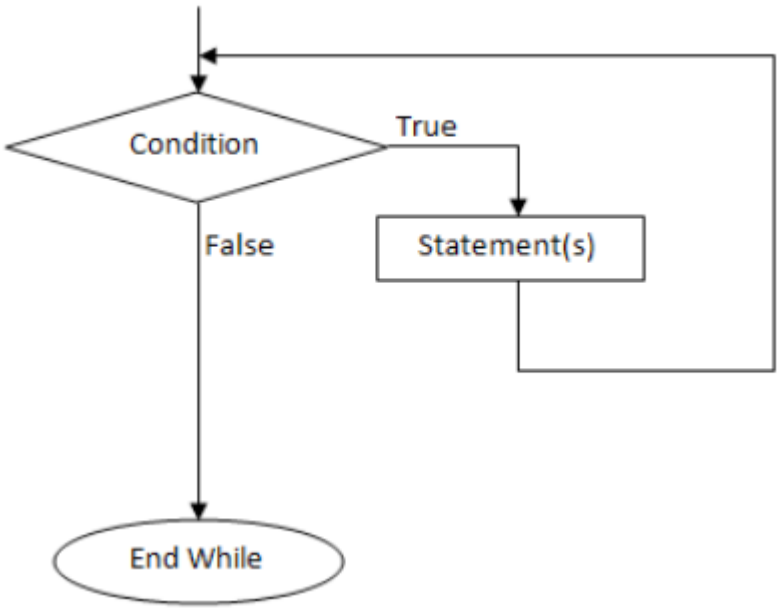
```
    switch(alphabet)
    {
        case 'a':
            printf("Alphabet a is a vowel.\n");
            break;
        case 'e':
            printf("Alphabet e is a vowel.\n");
          break;
        case 'i':
            printf("Alphabet i is a vowel.\n");
          break;
        case 'o':
            printf("Alphabet o is a vowel.\n");
            break;
        case 'u':
            printf("Alphabet u is a vowel.\n");
            break;
        default:
            printf("You entered a consonant.\n");
    }
    return 0;
}
```

`break` Statement

The break statement is used to break out of a loop or a switch case. It is very important to use break statement inside switch case, otherwise when a matching case is found, all the cases below it are executed by default. So, break statement is used after each case in order to break out of switch case after a case has been matched.

12. `while` loop in C

While loop is an entry controlled loop i.e. the condition is checked before entering into the loop. So if the condition is false for the first time, the statements inside while loop may not be executed at all. When the condition becomes false, the program control exits the loop. We can also exit a loop by using break statement like in switch case.



Example 14

C program to print the multiplication table of 2 from 1 to 10.

```
#include<stdio.h>
int main()
{
    int i=1;
    while(i<=10)
    {
        printf("2 * %d = %d\n",i,2*i);
        i++;
```

```
    }
    return 0;
}
```

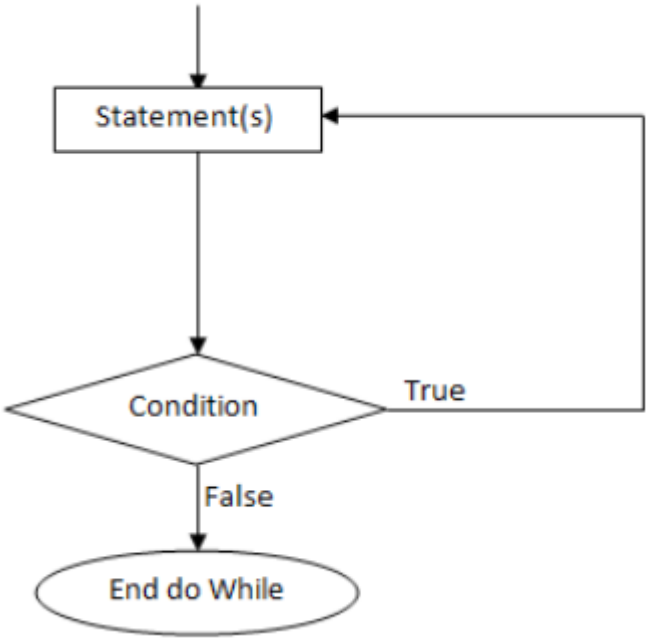
Example 15

Infinite while loop

```
#include<stdio.h>
int main()
{

    while(1)
    {
        printf("This is infinite loop");
      //break; // Uncomment break statement to break out of infinite
loop
    }
    return 0;
}
```

13 do-while loop in C

Do-while loop is an exit controlled loop i.e. the condition is checked at the end of loop. It means the statements inside do-while loop are executed at least once even if the condition is false. Do-while loop is an variant of while loop. In order to exit a do-while loop either the condition must be false or we should use break statement.



Example 16

C program to print the multiplication table of 5 from 1 to 10.

```
#include<stdio.h>
int main()
{
    int i=1;
    do
    {
        printf("5 * %d = %d\n",i,5*i);
        i++;
    }while(i<=10);
    return 0;
}
```

Example 17

Infinite Do while loop

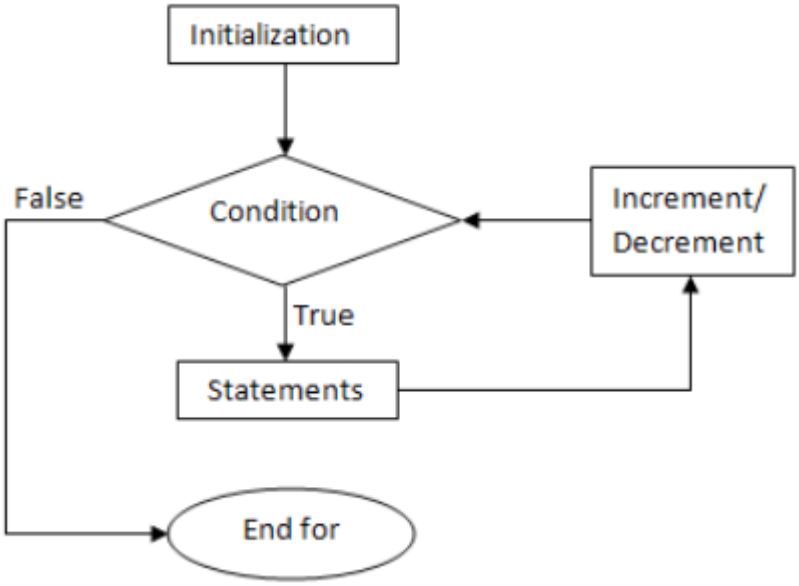
```
#include<stdio.h>
int main()
{
    do
    {
        printf("This is infinite loop");
      //break; // Uncomment break statement to break out of infinite Do while loop
    }while(1);

    return 0;
}
```

14.For loop

For loop is an entry controlled loop i.e. the condition is checked before entering the loop. So if the condition is false for the first time, the statements inside while loop may not be executed at all. To exit from a for loop, either the condition should be false or a break statement should be encountered. For loop is suitable to use when we have to run a loop for a fixed number of times

Syntax

```
for (initialization; condition; increment/decrement)
{
    statement(s);
    ... ... ...
}
```



Example 18

C program to find the sum of first n natural numbers.

```
#include<stdio.h>
int main()
{
    int i,n,s=0;
    printf("Enter value of n:");
    scanf("%d",&n);
    for(i=1;i<=n;i++)
    {
        s=s+i;
    }
    printf("Sum = %d",s);
    return 0;
}
```

Example 19

Infinite for loop

```
#include<stdio.h>
int main()
{
    for (int i=0; i>0; i++){
      printf("This is infinite loop");
     }

    return 0;
}
```

Example 20

Infinite loop with no conditions

```
#include<stdio.h>
int main()
{
    for (int i=0;; i++){
      printf("This is infinite loop");
     }

    return 0;
}
```

Example 21

Break statements in Infinite loop

```
#include<stdio.h>
int main()
{
    for (;;){
      printf("This loop will run only once");
      break;
     }

    return 0;
}
```

1. Write a C program to compute the roots of the equation $ax^2+bx+c= 0$ and print using three-decimal places. The roots are real $(-b\pm\sqrt{D})/2a$, if the discriminant $D=b^2-4ac$ is non-negative. If the discriminant is negative, then the roots are complex conjugate $(-b/2a)\pm((\sqrt{-D})/2a)i$.

2. A positive decimal fraction can be expressed in binary system as $0.x_1x_2x_3x_4\cdots$, where x_i's are either zero or one. Write a C program that accepts (from the keyboard) a positive decimal fraction a($0< a <1$) and prints out at most first four bits of the equivalent binary representation. If the binary representation continues after four bits, then it append the binary representation with ···

3. Write a C program that reads a real number x from the keyboard and calculates the sum of the series

$$\sum_{n=0}^{\infty} tn,\ tn= (-1)^n * (x^{2n+1})/(2n+ 1)!$$

by adding terms as long as $|t_n|>10^{-6}$.

4. Write a C program that accepts a positive integer from the keyboard and checks whether the entered number is a perfect number.

5. Write a C program that accepts a sequence of positive integers between 1 and 9 both inclusive from the keyboard. The program will stop accepting input once an integer outside the range is entered. The program will finish by printing the total number that are multiples of 2.