KATHMANDU UNIVERSITY

Department of Computer Engineering

A

Lab Report On

Computer Programming {COMP 102}

Lab Sheet No: 3

Submitted by:

Ashraya Kadel

UNG CE I/I

Roll No: 25

Submitted to:

Sameer Tamrakar

Department of Computer Engineering.

WEEK 6: FUNCTION

In week 6, we learnt about the use of functions in C. We did some common programs using functions.

<Q.1> Write a program to identify whether the given number is perfect or not using a function.

Ans:

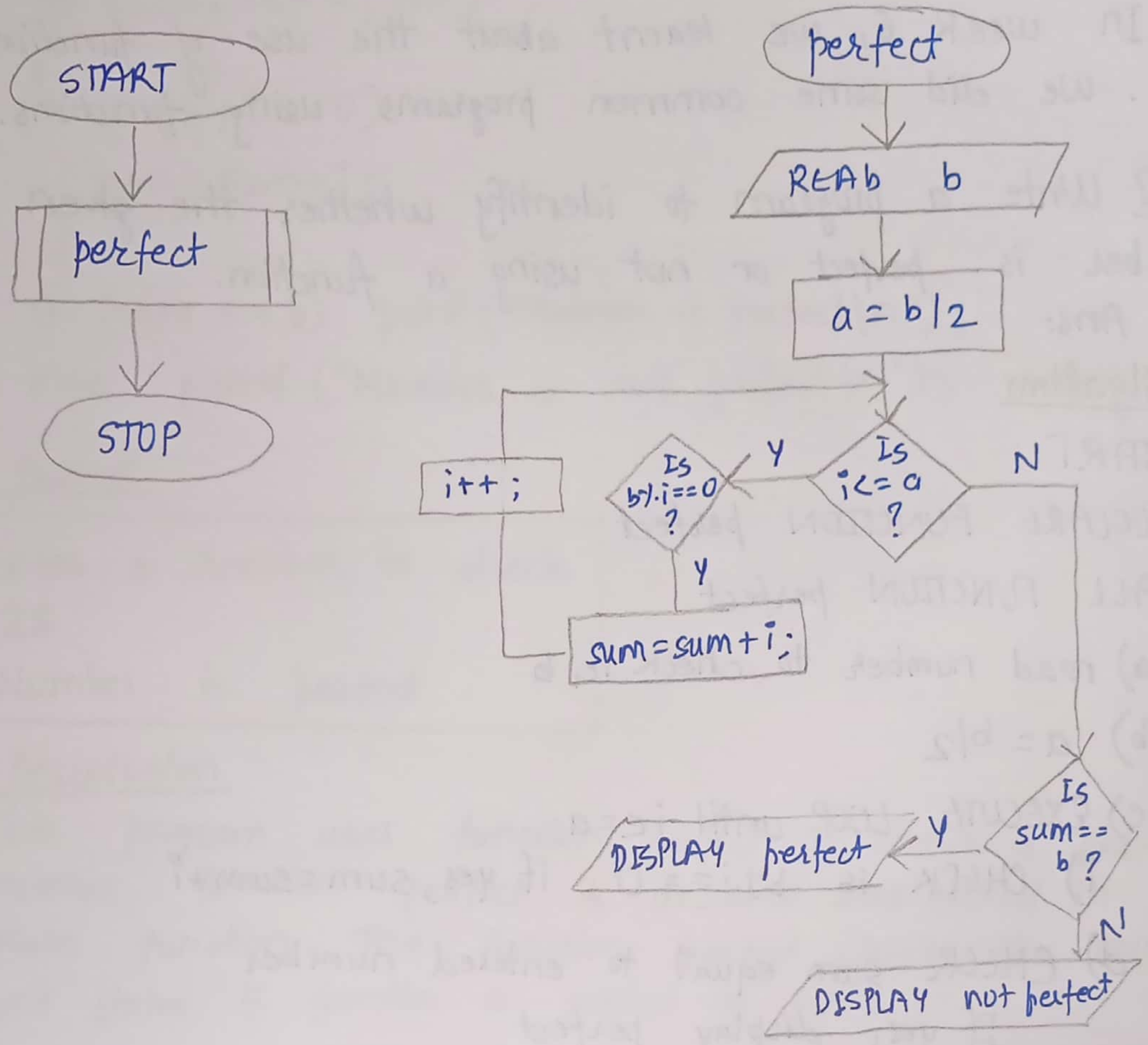
*) Algorithm

```
A) START
B) DECLARE FUNCTION perfect
C) CALL FUNCTION perfect
    a) read number to check ie, b
    b) a = b/2
    c) EXECUTE LOOP until i<=a
        i) CHECK is b%i == 0. if yes, sum=sum+i
    d) CHECK sum equal to entered number
            If yes, display perfect
            If no, display not perfect.
```

*) Source code

```
#include <stdio.h>
void perfect (void);
void main ( )
{
  perfect ();
}
void perfect ( )
{
  int a, b, sum=0, i;
  printf ("Enter no. to check\n");
  scanf ("%d", &b);
```

*) Flowchart


START
perfect
STOP
perfect
READ b
a = b/2
Is i<= a ?
Y
N
Is b%i==0 ?
Y
i++;
sum=sum+i;
Is sum== b ?
Y
N
DISPLAY perfect
DISPLAY not perfect

```
a=b/2;
for (i=1; i<=a; i++)
{
    if (b%i==0)
    sum = sum+i;
}
if (sum == b)  printf("Number is perfect\n");
else  printf ("Number is not perfect \n");
```

* Output

```
Enter a number to check
28
Number is perfect
```

*) Description

This program uses function perfect to check a perfect number. function perfect is declared and called in main function. The function perfect checks the number and states if function is perfect or not.

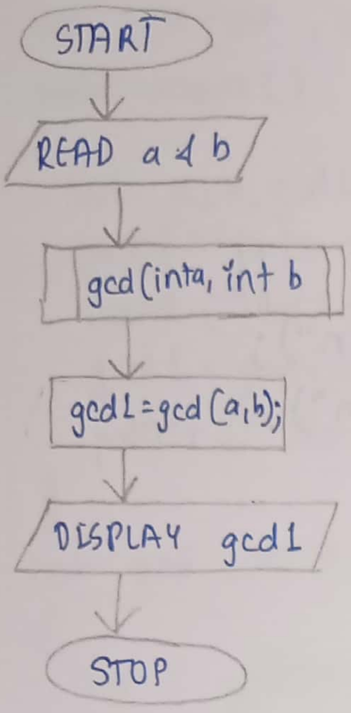
<Q.27>: Write a program to evaluate GCD of two given integers.

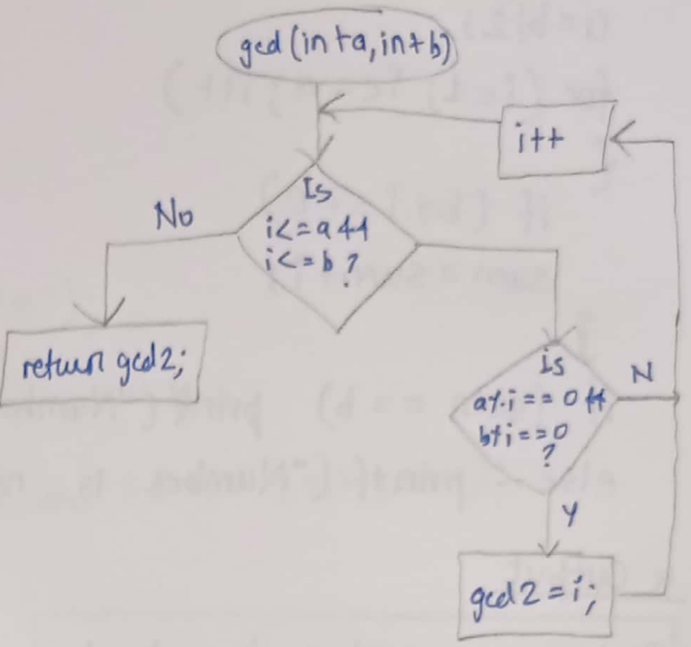
Ans:

*) Algorithm

```
i) START
ii) DECLARE function gcd (int, int)
iii) READ two numbers a & b.
iv) CALL function gcd (a, b)
    A) EXECUTE LOOP until i<=a && i<=b
        a) CHECK a%i==0 && b%i==0, if yes gcd2=i;
    B) return gcd2;
v) gcd1 = returned gcd2 value from function gcd
vi) DISPLAY gcd1
vii) STOP
```

*) Flowchart


START
READ a & b
gcd(int a, int b
gcd1=gcd(a,b);
DISPLAY gcd1
STOP


gcd(int a, int b)
i++
Is i<=a && i<=b ?
No
return gcd2;
Is a%i==0 && b%i==0 ?
N
Y
gcd2=i;

(*) Source code

```
#include<stdio.h>
int gcd (int , int );
void main ()
{
  int a, b, gcd1;
  printf ("Enter two numbers\n");
  scanf ("%d%d", &a, &b);
  gcd1 = gcd (a, b);
  printf (" The GCD of %d and %d is = ", a, b, gcd1);
}
int gcd (int a, int b)
{
  int i, gcd2;
  for (i=1; i<=a && i<=b; i++)
  { if (a%i==0 && b%i==0) gcd2=i;
  }
  return gcd2;
}
```

(*) Output

```
Enter two numbers
366
60
The GCD of 366 and 60 is = 6
```

(*) Description

This program uses function to check GCD between two number.
Here, the function gcd checks for gcd between a and b and then returns the gcd value to main function and displays it.

<Q.3> WAP to reverse a given number.

Ans:

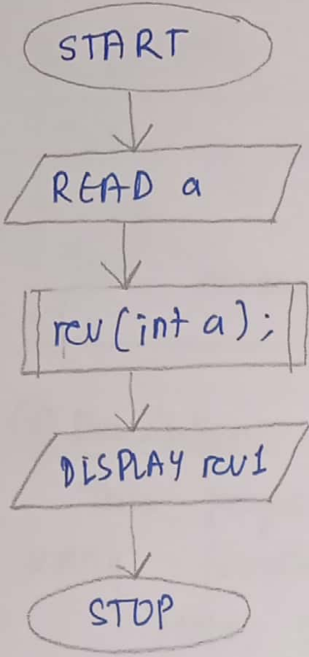
⊛ Algorithm

```
i) START
ii) DECLARE function rev(int )
iii) READ a
iv) CALL function rev (a)
    A) EXECUTE loop while a!=0
        a) c=a%10
        b) rev2=rev2*10+c
        c) a=a/10
    B) return rev2
v) rev1= return rev2 value from function rev
vi) DISPLAY rev1
vii) STOP
```

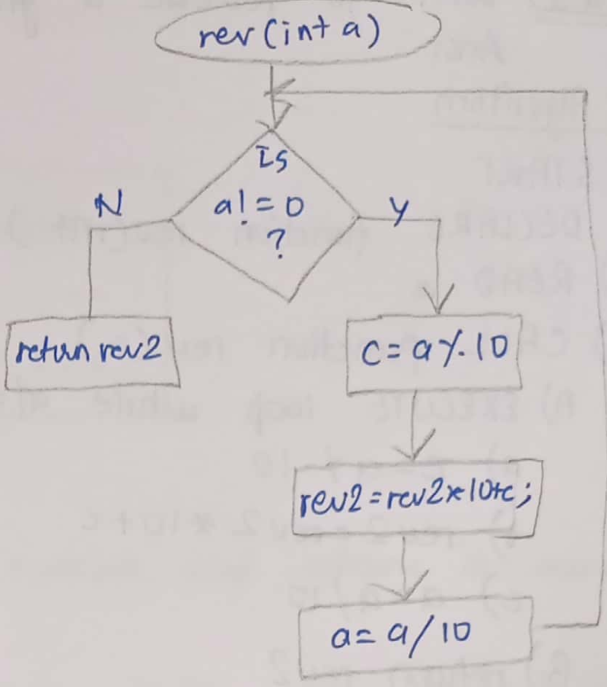
⊛ Source Code

```
#include <stdio.h>
int rev (int );
void main ( )
{
  int a, rev1;
  printf ("Enter a number to reverse\n");
  scanf ("%d", &a);
  rev1 = rev (a);
  printf ("The reverse of number %d is %d", a, rev1);
}
int rev (int a)
{
  int rev2=0, c=0;
  while (a!=0)
  {
     c=a%10;
     rev2 = rev2*10+c
     a=a/10
  }
```

(*) Flowchart



START
READ a
rev(int a);
DISPLAY rev1
STOP



rev(int a)
Is a!=0 ?
N
Y
return rev2
C=a%10
rev2=rev2*10+c;
a=a/10

```
    return rev2;
}
```

⊛ Output

```
Enter a number to reverse
155
The reverse of 155 is 551.
```

⊛ Description:

This program reads a number and returns its revese using function rev.
Here, 155 is read in main and passed to function rev. 155 is reversed to 551 and the rev2 value is retuned to rev1 in main function which is displayed.

WEEK 7 : FUNCTIONS

In week 7, we continued with functions and started to do tougher programs using them.

<Q.1> WAP to find prime numbers from 1 to 100.
Ans:

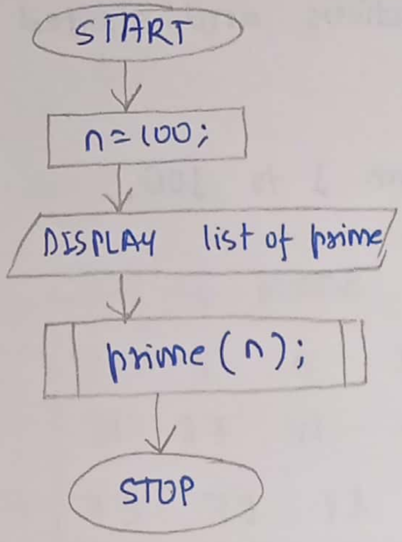
Algorithm

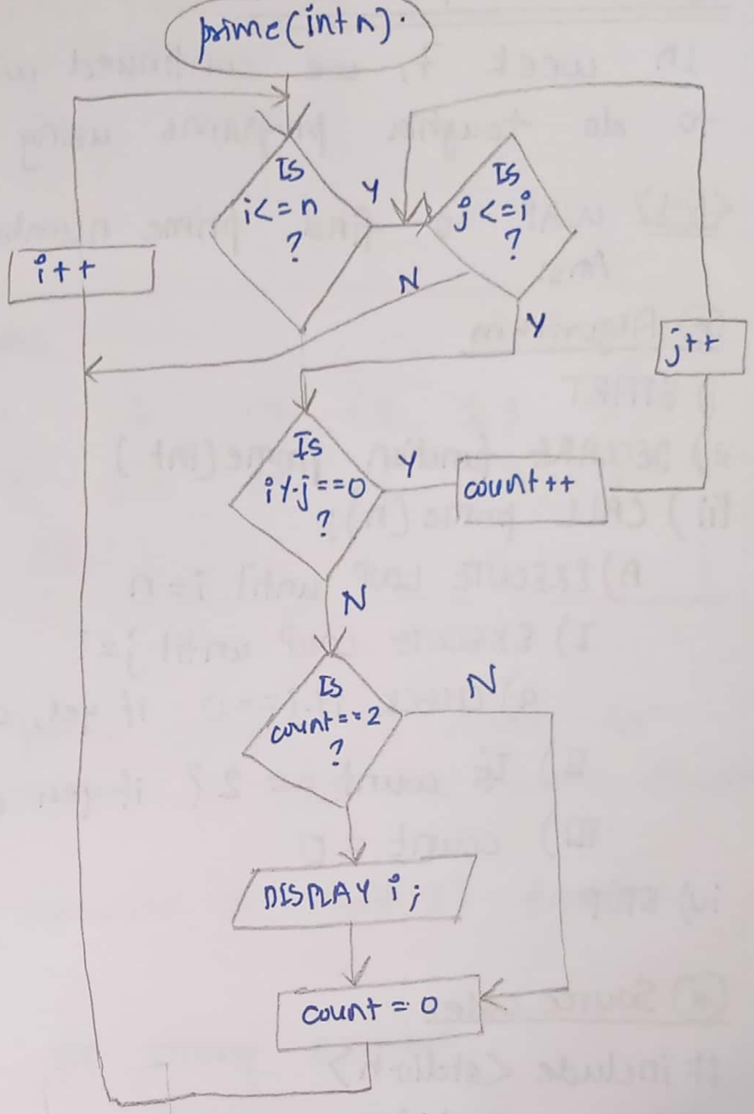
```
i) START
ii) DECLARE function prime(int )
iii) CALL prime(n);
    A) EXECUTE LOOP until i=n
        I) EXECUTE LOOP until j=i
            a) CHECK i%j==0 if yes, count=count+1
        II) If count == 2? if yes, display i
        III) count = 0
iv) STOP
```

Source code:

```
#include <stdio.h>
void prime (int );
void main ( )
{
  int n=100;
  printf ("List of prime numbers till 100 ~~");~~ \n");
  prime(n);
}
void prime (int n)
{
  int i, j, count=0;
  for (i=1; i<=n; i++)
  {
    for (j=1; j<=i; j++)
    {
      if (i%j==0) count = count+1;
    }
```

⊛ Flowchart


START
n=100;
DISPLAY list of prime
prime(n);
STOP


prime(int n).
Is i<=n ?
Y
N
Is j<=i ?
Y
i++
j++
Is i%j==0 ?
Y
count++
N
Is count==2 ?
N
DISPLAY i;
count=0

```
    if (count == 2)  printf("%d\t", i)
    count = 0;
   }
}
```

⊛ Output

```
List of prime numbers
2    3    5    7    11   13   17   19   23   29
31   37   41   43   47   53   59   61   67   71
73   79   83   89   97
```

⊛ Description:

This program uses prime function to check for prime numbers and displays it from the prime function itself.

Here, no value is returned to main () function.

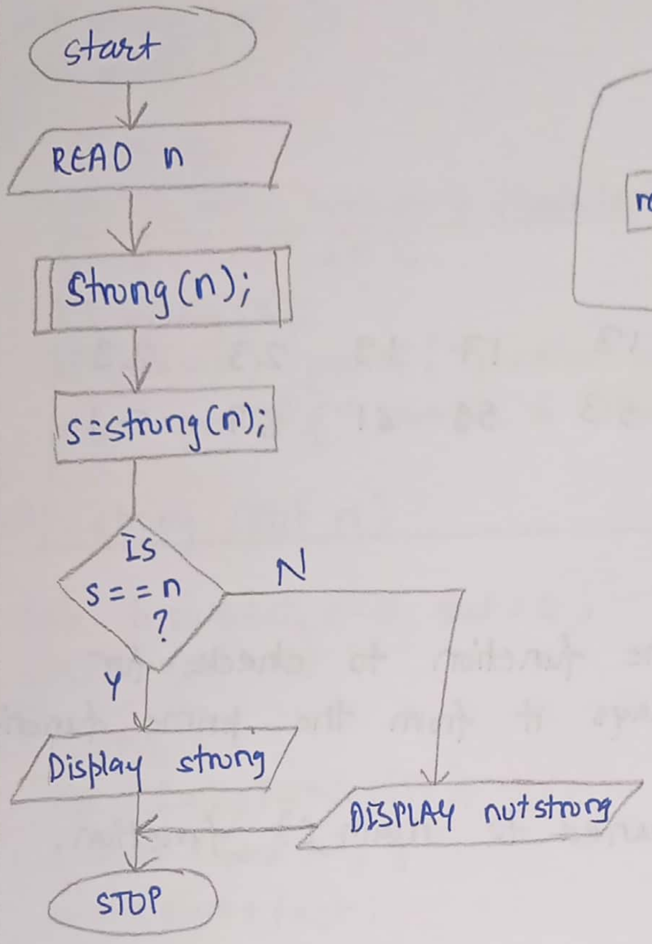
<Q.2> WAP to check for strong number.

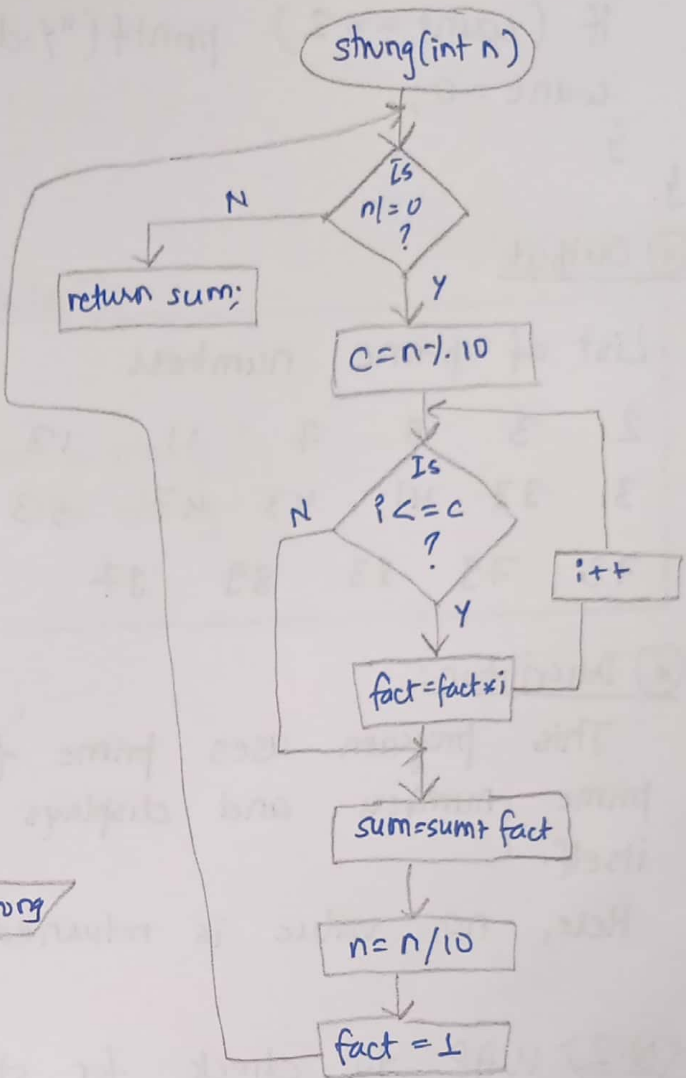
Ans:

*) Algorithm

```
i) START
ii) DECLARE function strong(int)
iii) READ n
iv) CALL function strong(n)
   A) EXECUTE loop until n != 0
      a) c = n%10
         ~~I) fact = fact * i~~
      b) EXECUTE LOOP until i = c
         I) fact = fact * i ;
      c) sum = sum + fact ;
      d) n = n/10
      e) fact = 1
   B) return sum.
v) s reads value of sum
   returned from function
   strong
vii) CHECK (s == n)
     If yes, display strong
     If no, display not strong
```

(*) Flowchart


start
READ n
Strong(n);
s=strong(n);
Is s==n ?
N
Y
Display strong
DISPLAY notstrong
STOP


strong(int n)
Is n!=0 ?
N
return sum;
Y
c=n%10
Is i<=c ?
N
Y
fact=fact*i
i++
sum=sum+fact
n=n/10
fact=1

⊛ Source code:

```
#include <stdio.h>
int strong (int );
void main ( )
{
    int n,s;
    printf("Enter number to check \n");
    scanf ("%d", &n);
    s= strong (n);
    if (s==n)  printf ("Number is strong");
    else  printf (" Number is  not strong");
}
int  strong  (int n)
{
   int  i, sum=0, c=0, fact=1 ;
   while  (n!=0)
   {
     c=n%10;
     for (i=1;i<=c;i++)
     { fact = fact *i; }
     sum = sum+fact ;
     n=n/10;
     fact = 1;
   }
     return  sum;
}
```

⊛ Output

```
Enter  number  to  check
145
Number  is  strong
```

⊛ Description

This program reads the number from uses and checks whether the number is strong or not.

Here,
function strong (int n) returns the sum of factorial of each digit of entered numbes which is checked in main function and the result is displayed.