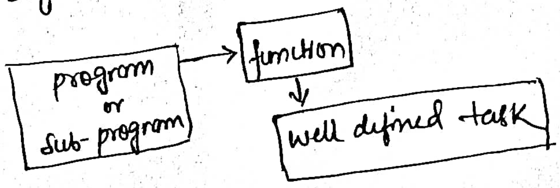
DSA- class- 6   5/02/23 (sunday)

Functions :

Linked with a well-defined task



so, function is a program which is linked to a well defined task.

To print babbar 5 times

```
for (int i=0; i<5; i++){
        cout <<" babbar" <<endl;
   }
```

for user input : replace 5 by n

To print babbar 5 times we have to repeatedly write the same code so it will make the code bulky and lengthy and if there is mistake in code then it will appear in all codes so it will become buggy

To overcome above mentioned drawbacks we use functions

```
void printname () {
        int n;
        cout <<"Enter n " <<endl;
        cin >>n;
        for (int i=0; i<n; i++){
          cout << "babbar" <<endl;
        }
   }
int main () {

    cout << printname() <<endl;

    return0;
  }
```

Syntax

return type function name (parameter) ← Input
{
  // function body → work of function, → return type
}

Explanation

$\text{return type} \leftarrow$ int main $\uparrow$ function ( ) → no input parameter
{
- - -
- - -
- - -
- - -
} function body

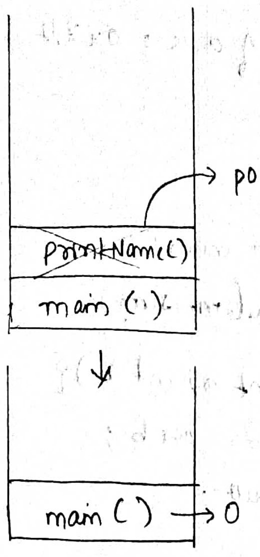
return 0; → main function successfully execution
}

function
↓
well-defined task
↓
Reuse + refer
↓
readable

Function call stack

stack
↳ Data structure
↳ Data stored in specific way
↳ LIFO - Last in first out

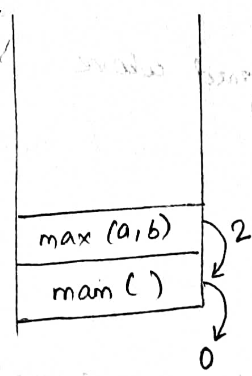
eg¹



→ function call
→ kis function ne dusre function ko call kiya hai
→ function k kaun kaun se local variable hai
→ function kya return kar rha hai

```
main ( )
{  ①
   ↓
   print Name ( )      ②  →   printName ( )
   return 0;                  {
}                               int n
          ③  ←                  cin >> n;
                                for (      ) {
                                  cout << " " ;
                                }
                              }
```

eg2



```
main ( )
{
① → int a = 1 ;
  ↳ int b = 2 ;
  → cout << max (a, b);     ②  →  int max ( int a, int b)
⑤ } return 0;                     {
              ↓  ↓ ↓                if (a>b)   (1 > 2) ×
              1  2                    return a;          ③
        function call              else
                    ④  ←             return b;
                                  }
```

main function is returning 0 to operating system

Note

function call & function invoke means same

Pass by Value

Pass by value
↳ copy create karna

```
main (  )
{
 int num = 12; →  memory mai ek block
                  banjaega of num → value 12
 num++;
  ++ num; → [14]
 print (num)  ───────────→  void print (int num)
 --num; → [13]                              ↑
 cout << num ;                        copy creation
}                            {
                               ++num;  →  [14]
                                            ↓
                               num++;  →  [15]
                                            ↓
                               cout << num; → [16]
                                            ↓
                               -- num;  →  [15]
```

value of num at main ( ) & print (int num) are different

print (int num) is copy of num of main ()

To find address of a variable

```
int a = 5;
cout << " Address of a is : " << &a << endl;
```

Output

```
Address of a is : 0x7ff....
```

function to add 2 numbers

```
int main () {
int a, b;
cout << " Enter a " << endl;
cin >> a;
cout << " Enter b" << endl;
cin >> b;
int sum = add (a, b);
cout << " Addition ~~result is~~ : " << sum << endl;
return 0;
}
```

always above main ()

```
// function declare + define
int add (int a, int b){
  int result = a+b ;
  return result;
}
```

Note

when a function is invoke then that function should be declared above the ~~ma~~ function where it is called or invoked.

find max of 3 numbers

```
int main () {
 int a, b, c ;
 cin >> a >> b >> c;
 int maximum = findmax (a, b, c);
 cout << maximum << endl;
 return 0.
}
```

```
int findmax(int n1, int n2, int n3)
{
  if ( n1 > n2 && n1 > n3)
    return n1;
  else if ( n2 > n1 && n2 > n3)
    return n2;
  else
    return n3;
}
```

To count numbers

```
int main () {
int n;
cin >> n;
print counting (n);
return 0;
}
```

```
void print counting (int n) {
  for (int i = 0; i <= n; i++) {
    Cout << i << " ";
  }
  cout << endl ;
}
```

Student Grade Problem

```
/* int marks ;
cout << "Enter mark" <<endl;
cin >> mark ; */

for (int i=1; i<=100 ; i++){
    char ans = getGrade(i);
    cout << " Grade for mark="<< i << ans <<endl;
}
```

```
switch (marks/10) {
    case 10: return 'A'; break;
    case 9: return 'B'; break;
    case 8 : return 'C' ; break;
    case 7 : return 'D' ; break;
    case 6 : return 'E';
}
```

Sum of even no. upto N

```
N= 10
1  2  3  4  5  6  7  8  9  10
   ↓     ↓     ↓     ↓     ↓
   2  +  4  +  6  +  8  + 10  = 30
```

```
int n;
cout<< " Enter n" <<endl;
cin>>n;
int ans = getEvensum(n);
cout << " Evensum is"<< ans <<endl;
```

```
int getevensum(int n) {
    int sum =0;
    for (int i=2; i<=n;i=i+2) {
        sum = sum+i ;
    }
    return sum;
}
```