CS & IT ENGINEERING

Programming in C

Functions & Storage Classes

Lec- 02

By- Pankaj Sharma sir

TOPICS TO BE COVERED

- Functions-2

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

        printf("Hello");
    }
```

```
#include<stdio.h>
void main(){
        int i;
    i = printf("Hello");
    printf("%d",i);
        }
```

```
#include<stdio.h>

void main(){

      int a=10,b=20;

      mul(a,b);

         }
```

```
int mul(int x, int y)
   {
      int temp;

      temp = x*y;

      return temp;
   }
```

```c
#include<stdio.h>

   void mul(int,int);


void main(){

      int a=10,b=20;

        mul(a,b);

      }
```

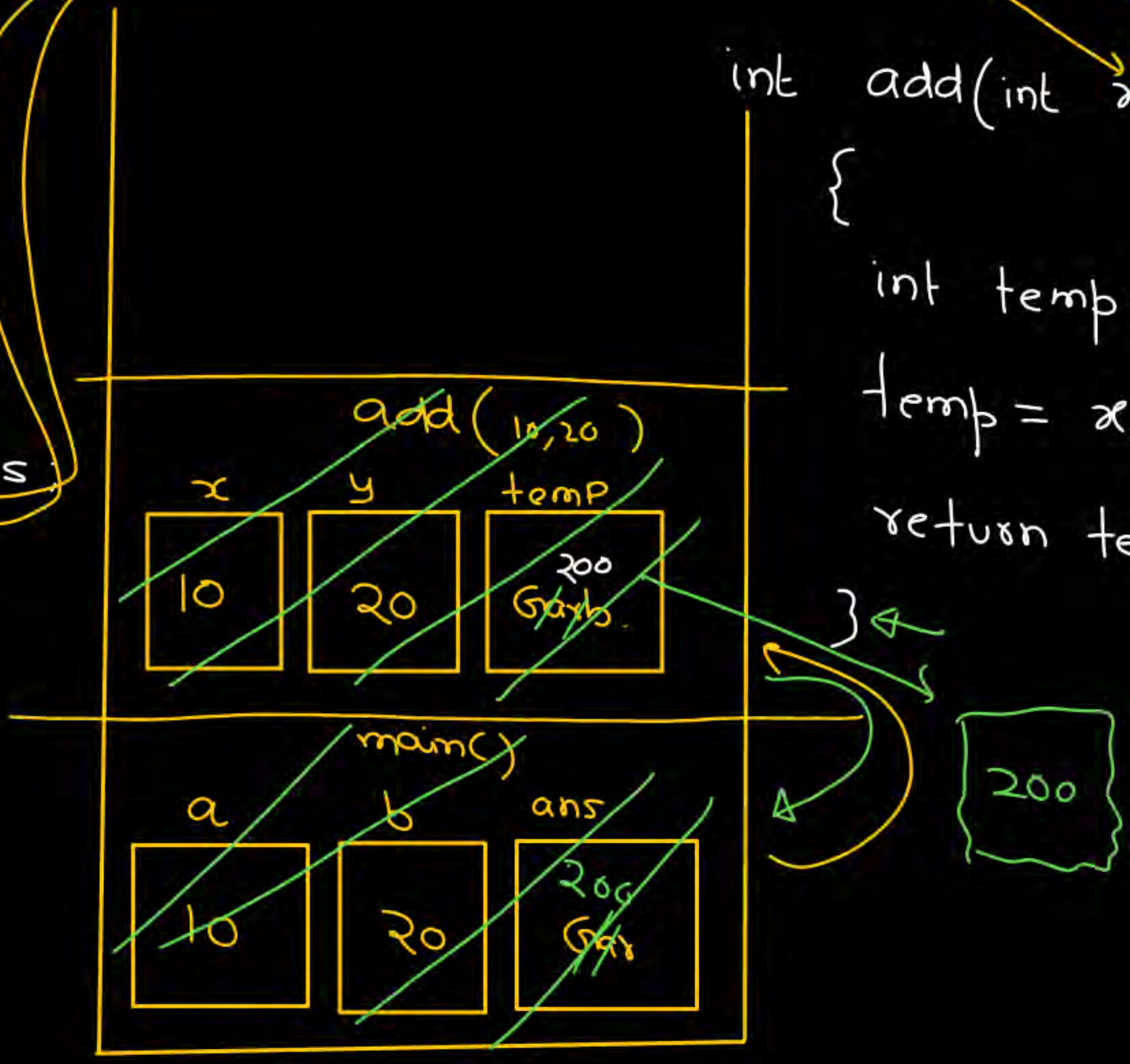
```c
void mul(int x,int y)
 {
   int temp;

   temp = x*y;

   printf("%d", temp);
 }
```

How a function works

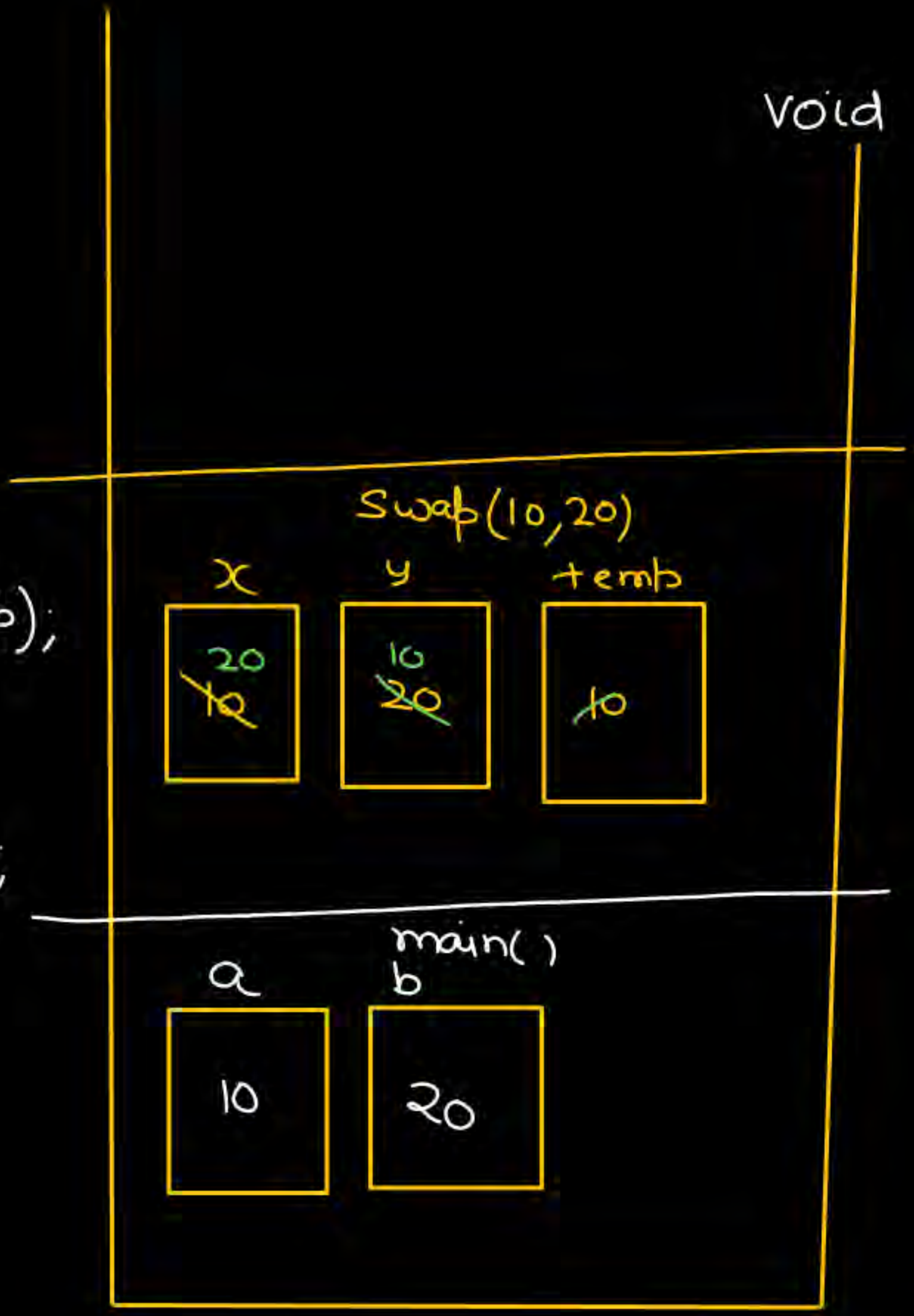
```
#include<stdio.h>
int add(int, int);
void main(){
✓   int a=10,b=20,ans;
✓   ans = add(a,b);
✓   printf("%d",ans);
}
```

200

```
int add(int x, int y)
{
  int temp;
  temp = x*y;
  return temp;
}
```



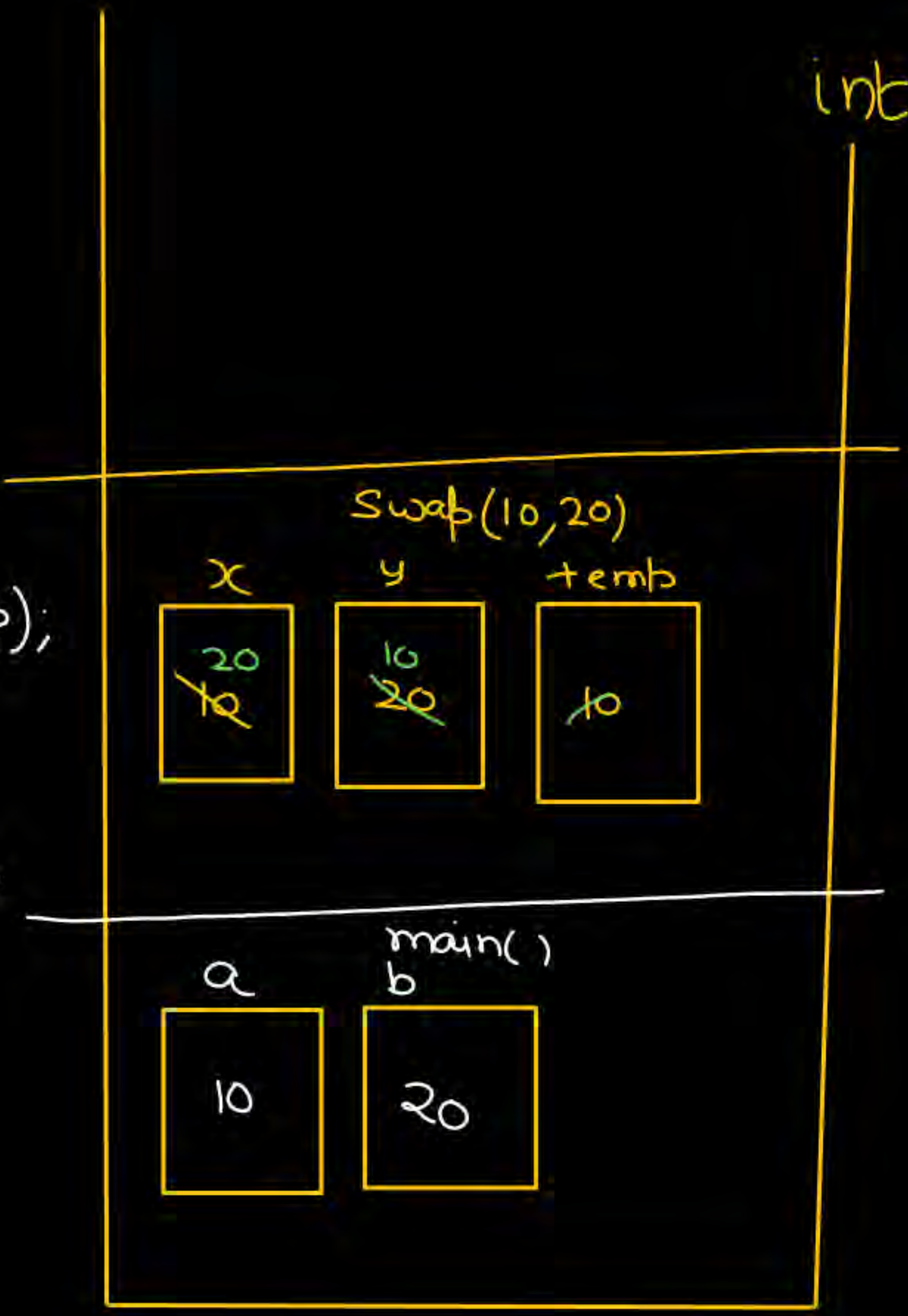
```
#include<stdio.h>
 void swap(int ,int);
  void main( ){
     int a=10, b=20;
printf(" a = %d , b= %d" ,a,b);
     swap(a,b);
printf("a= %d , b= %d",a,b);
          }

   a=10, b=20
```



```
void swap(int x, int y)
  {  int temp;
 temp = x;
  x = y;
 y = temp;
}
```

```
#include<stdio.h>
void swap(int ,int);
void main(){
    int a=10,b=20;
printf("a= %d , b= %d",a,b);
         10 20
    swap(a ,b);
printf("a= %d , b= %d",a,b);

        }
```

a=10,b=20



```
                10      20
int swap(int x , int y )
 {  int temp;
temp = x ;
 x = y ;
 y = temp ;
}
```

```
void main() {

}
```

```
int main() {

}
```

```
for(i=1; i<=5; i++)
{
    printf("Pankaj");
    printf("\n");
}
```

```
Pankaj
Pankaj
Pankaj
Pankaj
Pankaj
```

```
    Row
for(i=1; i<=4; i++)
    {

    pf("12345");
    pf("\n");
    }
```

Row	
1	12345
2	12345
3	12345
4	12345

```
for(Row=1; Row<=5; Row++)
    {
        different
        printing

    }
```

Row	
1	1
2	1 2
3	1 2 3
4	1 2 3 4
5	1 2 3 4 5

col

$$\begin{array}{cc|ccccc} & & 1 & 2 & 3 & 4 & 5 \\ \hline \text{Row} & 1 & 1 & & & & \\ & 2 & 1 & 2 & & & \\ & 3 & 1 & 2 & 3 & & \\ & 4 & 1 & 2 & 3 & 4 & \\ & 5 & 1 & 2 & 3 & 4 & 5 \end{array}$$

```
for(Row=1; Row<=5; Row++)
    {



    }
```

Row	col
1	1
2	1, 2
3	1, 2, 3
4	1, 2, 3, 4
5	1, 2, 3, 4, 5

```
for(Row=1; Row<=5; Row++)
    {
        for(col= 1 ; col<=Row ; col++)
            {


            }
    }
```

Row	last col
1	1
2	2
3	3
4	4
5	5

Row	col
1	1 ✓
2	✓ 1, $\textcircled{2}$
3	✓ 1, 2, 3
4	✓ 1, 2, 3, 4
5	✓ 1, 2, 3, 4, 5

col

$$\begin{array}{cc|ccccc} & & 1 & 2 & 3 & 4 & 5 \\ \hline \text{Row} & 1 & 1 & & & & \\ & 2 & 1 & 2 & & & \\ & 3 & 1 & 2 & 3 & & \\ & 4 & 1 & 2 & 3 & 4 & \\ & 5 & 1 & 2 & 3 & 4 & 5 \end{array}$$

```
for(Row=1; Row<=5 ; Row++)
   {
      for(col= 1 ; col<=Row ; col++)
           {
                  printf("%d",col);
           }
        printf("\n");
   }
```

$$
\begin{array}{ll|lllll|}
 & & \multicolumn{5}{c}{\text{col}} \\
 & & 1 & 2 & 3 & 4 & 5 \\
\hline
\text{Row} & 1 & 1 & & & & \\
 & 2 & 1 & 2 & & & \\
 & 3 & 1 & 2 & 3 & & \\
 & 4 & 1 & 2 & 3 & 4 & \\
 & 5 & 1 & 2 & 3 & 4 & 5 \\
\hline
\end{array}
$$

```
for(Row=1; Row<=5; Row+1)
      {

        for(col=1; Col<=Row; col++)
              {
                printf("%d", Row);
              }
            printf("\n");


     }
```

col

		1	2	3	4	5
Row	1	1				
	2	2	2			
	3	3	3	3		
	4	4	4	4	4	
	5	5	5	5	5	5

```
for(Row=1;Row<=5;Row++)
    {


    }
```

col

$$\begin{array}{c|ccccc}
\text{Row} & 1 & 2 & 3 & 4 & 5 \\
\hline
1 & 1 & 2 & 3 & 4 & 5 \\
2 & 1 & 2 & 3 & 4 & \\
3 & 1 & 2 & 3 & & \\
4 & 1 & 2 & & & \\
5 & 1 & & & &
\end{array}$$

```
for(Row=1;Row<=5;Row++)
    {

        for(col=1;col<=    ;col++)
            {


            }


    }
```

	col				
Row	1	2	3	4	5
1	1	2	3	4	5
2	1	2	3	4	
3	1	2	3		
4	1	2			
5	1				

Row	col
1	1, 2, 3, 4, 5
2	1, 2, 3, 4
3	1, 2, 3
4	1, 2
5	1

Row + last_val_of_col = 6

= 6 - Row

```
for(Row=1;Row<=5; Row++)
    {
        for(col=1 ; col<=6-Row ; col ++)
            {
                printf("%d",col);

            }
        printf("\n");
    }
```

Row	col_last value
1	5
2	4
3	3
4	2
5	1

Row	col
1	1, 2, 3, 4, 5
2	1, 2, 3, 4
3	1, 2, 3
4	1, 2
5	1

col

Row	1 2 3 4 5
1	1 2 3 4 5
2	1 2 3 4
3	1 2 3
4	1 2
5	1

```
12345
1234
123
12
1
```

```
for(Row=1;Row<= ~~5~~ n ; Row++)
    {
        for(col=1;col<= ~~6~~ (n+1)-Row; col++)
            {
            pf("%d",col);
            }
        pf("\n");
    }
```

i/p: n => 7

```
1234567
123456
12345
1234
123
12
1
```

```
for(Row=1; Row<=5 ; Row++)
    {




    }
```

```
        *
      * *
    * * *
  * * * *
* * * * *
```

```
for(Row=1; Row<=5 ; Row++)
    {

        In Every row

      i)    some space to be printed
     (ii)   some star to be printed

    }
```

```
        *
      * *
    * * *
  * * * *
* * * * *
```

```
for(Row=1; Row<=5 ; Row++)
   {

      In Every row

      i)   some space to be printed
      (ii) some star to be printed

   }
```

Row	spaces	star
1	4	1
2	3	2
3	2	3
4	1	4
5	0	5

```
Row
 1          *
 2        * *
 3      * * *
 4    * * * *
 5  * * * * *
```

Maths

```
for(Row=1; Row<=5 ; Row++)
{

    for( space=1; space<= 5-Row; space++)
            {
              printf(" ");
            }

    for( star =1; star<=Row; star++)
              {
                pf("*");
              }
    printf("\n");
}
```

Row

```
1         *
2       * *
3     * * *
4   * * * *
5 * * * * *
```

Row	spaces	star
1	4	1
2	3	2
3	2	3
4	1	4
5	0	5

spaces = 5-Row
star = No. of row Row

Maths

```
for(Row=1; Row<=~~5~~ n ; Row++)
    {

        for( space=1; space<= ~~5~~ n -Row; space++)
                 {
                   printf(" ");
                 }

        for( star =1; star<=Row; star++)
                   {
                     pf("*");
                   }
     printf("\n");
    }
```

```
Row
 1          *
 2        * *
 3      * * *
 4    * * * *
 5  * * * * *
```

Row	spaces	star
1	4	1
2	3	2
3	2	3
4	1	4
5	0	5

spaces = 5-Row

star = No. of Row Row

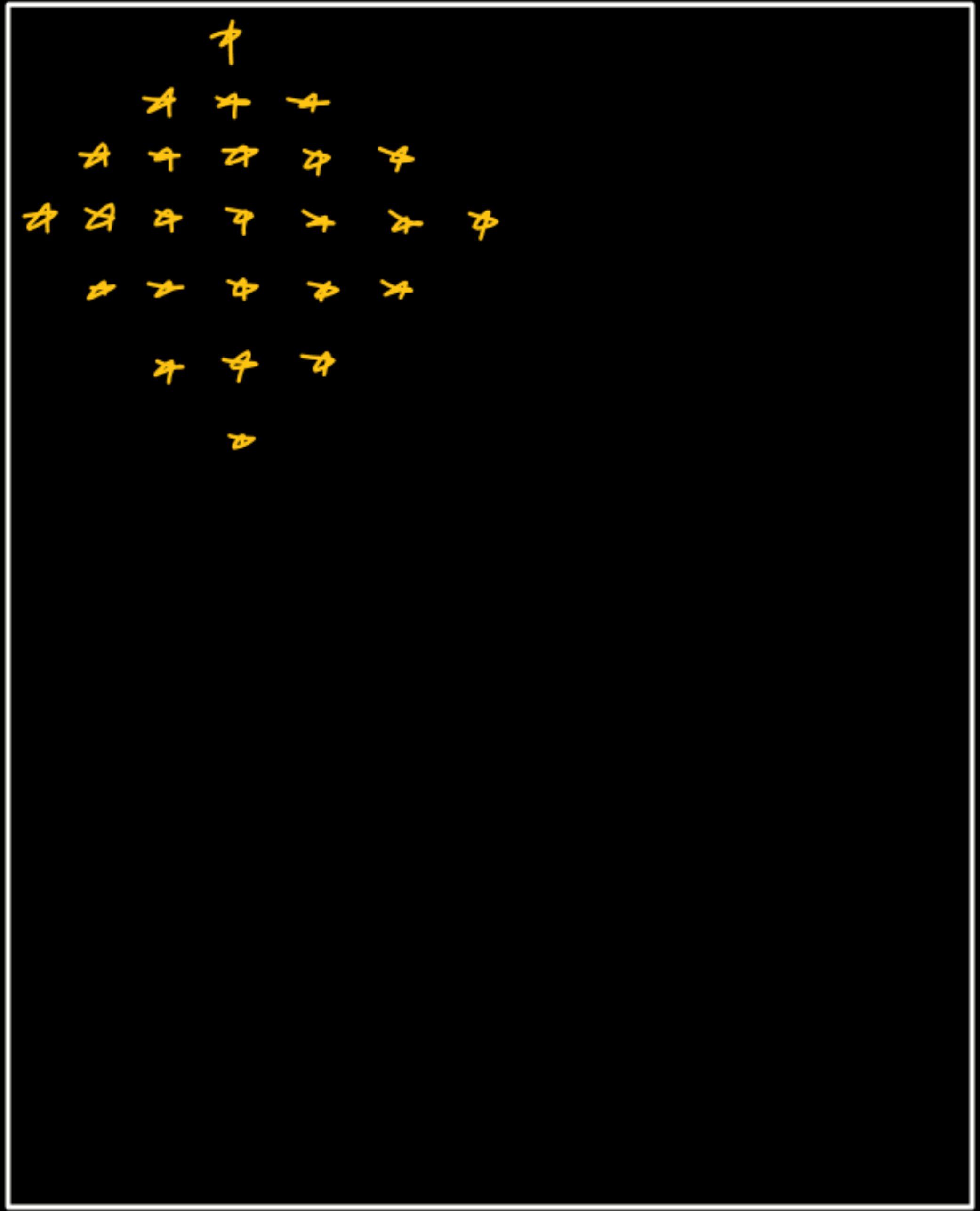
```
for (Row=1; Row<=4; Row++)
    {

    for( space=1; space<=4-Row; space++)
                pf(" ");

    for(star=1; star<=2*Row-1; star++)
                pf("*");

    printf("\n");
   }
```

```
1      *
2    * * *
3  * * * * *
4 * * * * * * *
```

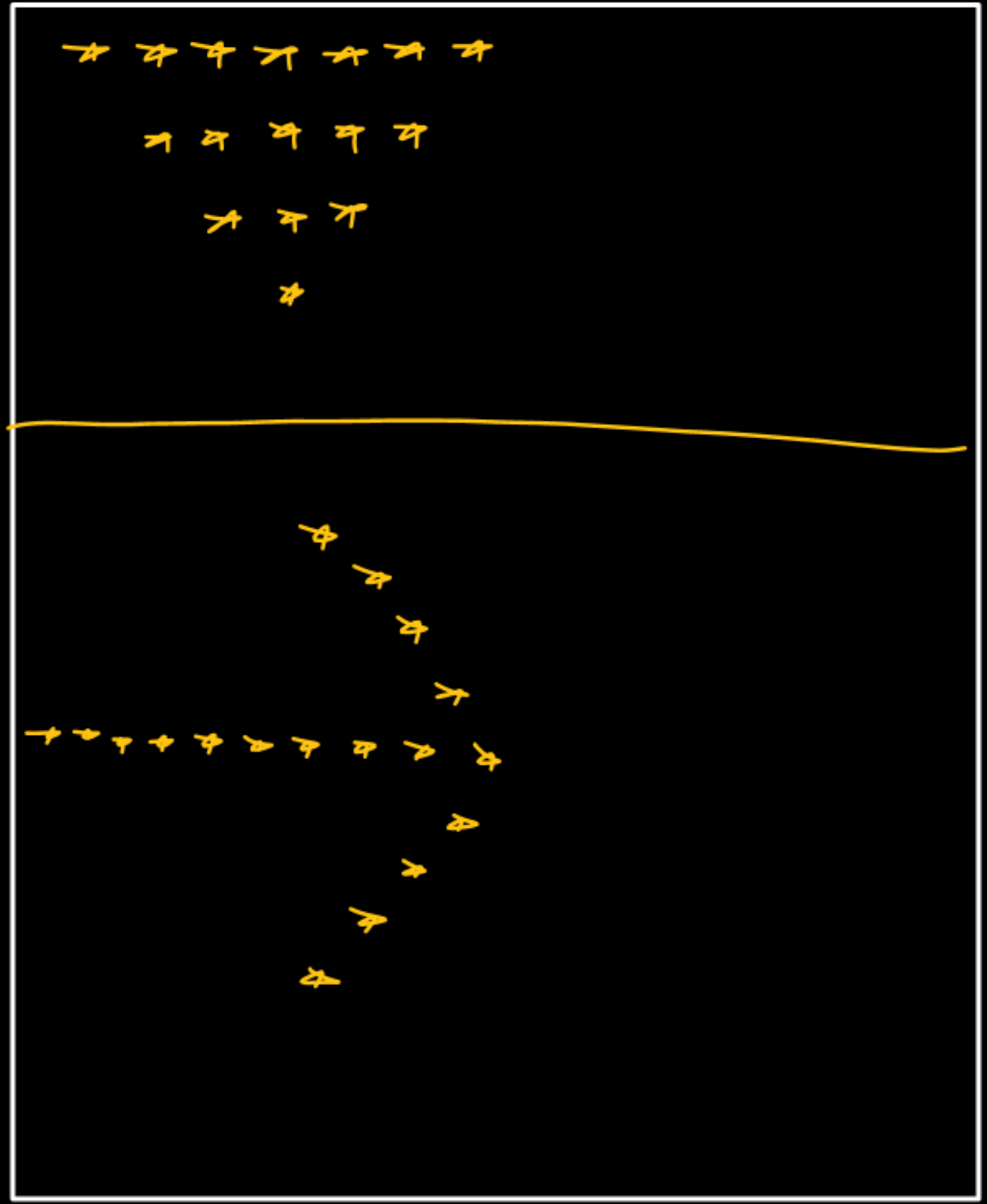
Row	space	star
1	3	1 (2*1-1)
2	2	3 (2*2-1)
3	1	5 (2*3-1)
4	0	7 (2*4-1)

for loop / logic

1.

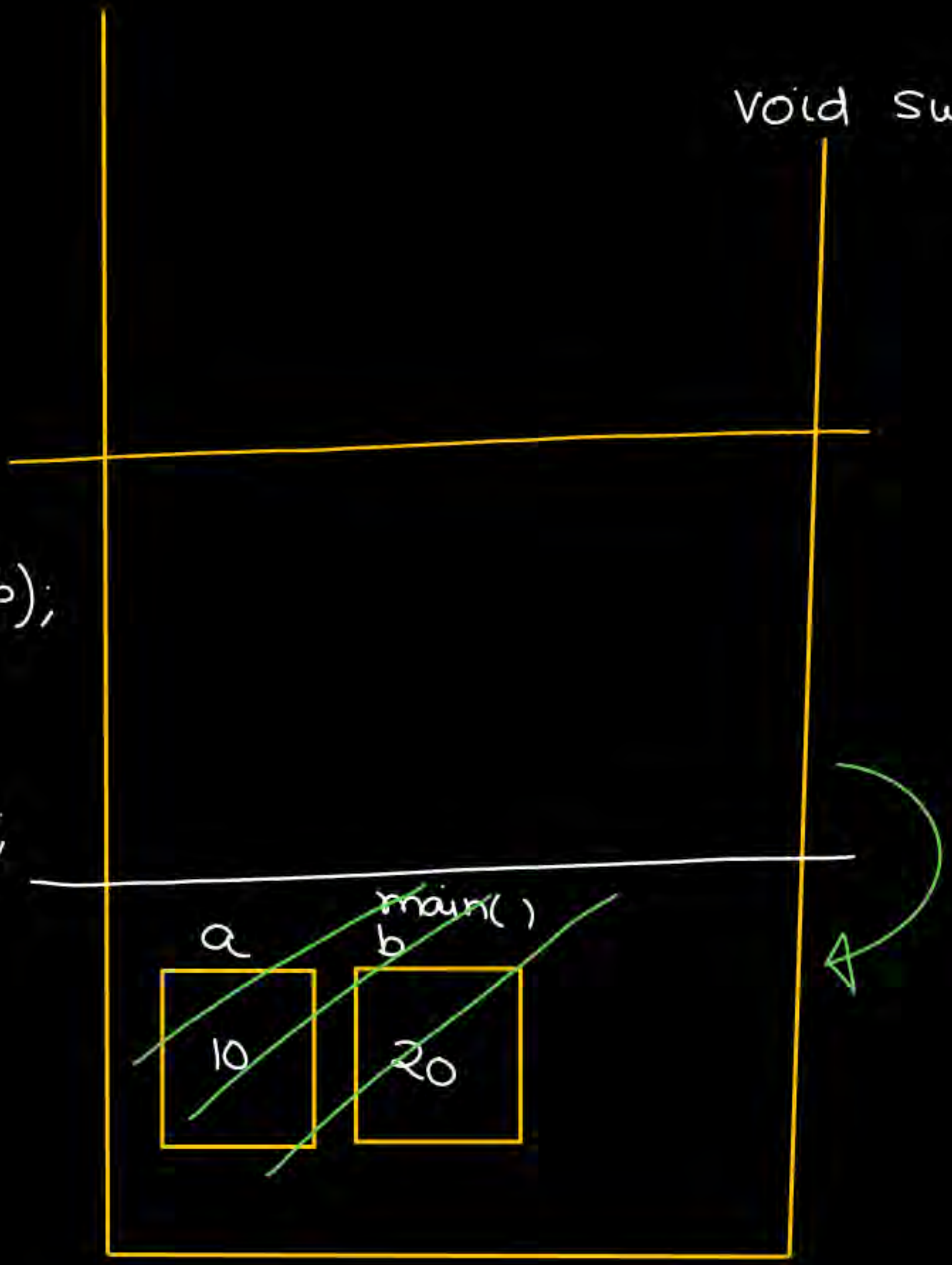


2.



```
#include<stdio.h>
 void swap(int ,int);
 void main(){
     int a=10, b=20;
printf("a= %d , b= %d",a,b);
     swap(a,b);
printf("a= %d , b= %d",a,b);

       }
```

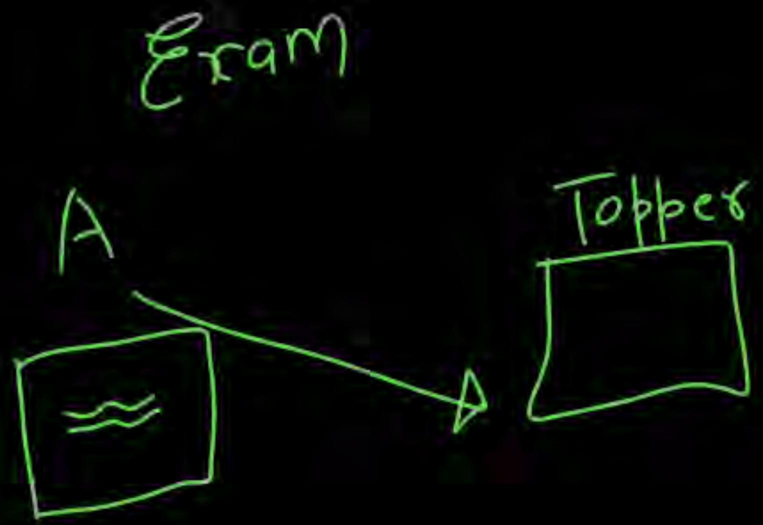
a=10, b=20
a=10, b=20



```
void swap(int x, int y)
   {  int temp;
   temp = x;
    x = y;
    y = temp;
   }
```

```c
#include<stdio.h>
void swap(int ,int);
void main(){
    int a=10,b=20;
printf("a = %d , b= %d",a,b);
                10 20
    swap(a,b);
printf("a = %d , b = %d",a,b);
    }
```

```
a=10,b=20
a = 10, b = 20
```



```c
                 10        20
void swap (int x , int y )
      { int temp ;
  temp = x ;
   x = y ;
   y = temp ;
  }
```

```
#include<stdio.h>

 void swap(int ,int);

  void main(){

     int a=10,b=20;
printf("a= %d ,b= %d",a,b);
     swap(a,b);
printf("a= %d ,b= %d",a,b);

          }

  a=10,b=20
  a=10,b=20
```

main call
करा रहा है
swap को

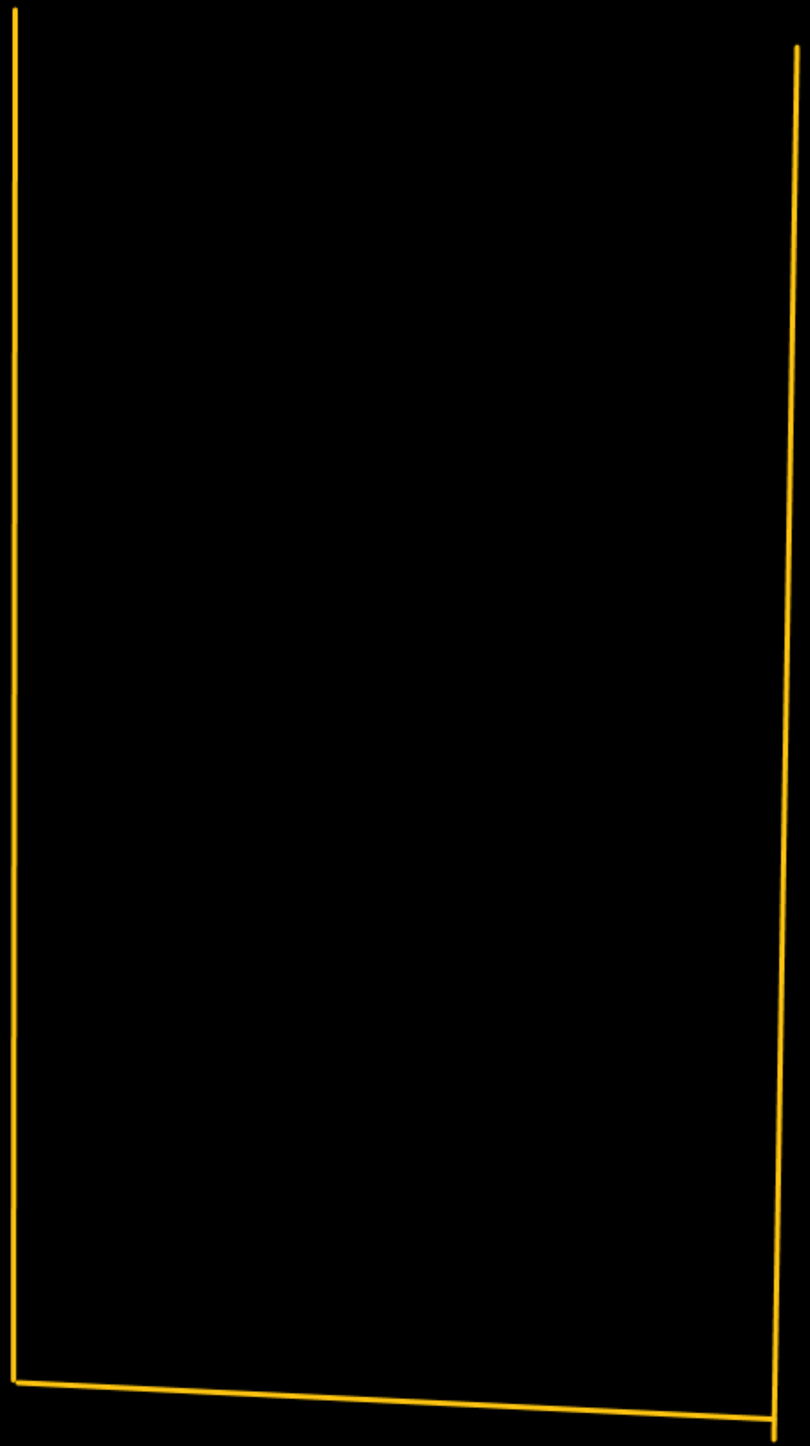
actual argument
actual parameter

formal Parameters

```
void swap(int x, int y)
  { int temp;
 temp = x;
  x = y;
  y = temp;
 }
```

10 20

```
# include<stdio.h>
void fun();
void main(){

        ______
        ______
        ______

        fun()
        ______
        ______
        ______

        }
```



```
void fun(){

        ______
        ______
        ______
        ______
        __

        }
```

```
main( ) {
    —
    —
    —

  swap(a, b);

    —
    —
  }
```

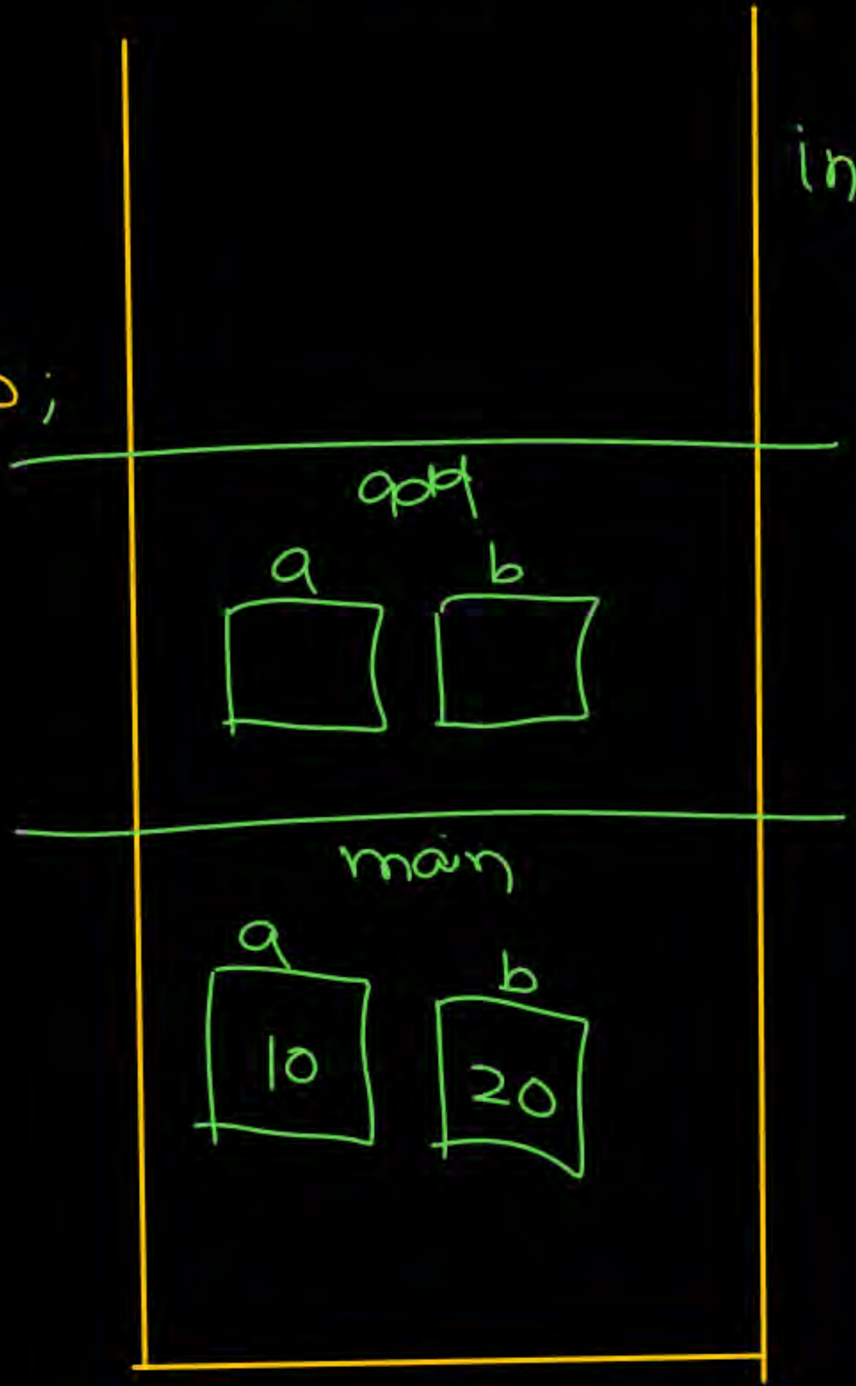
parameters are passed by value

swap

Call by value

Doubt ?

```
main() {
    int a = 10, b = 20;

      ≡

    add((a, b);

      =

  }
```



```
int add(int a, int b)
  {

  }
```

functions

printf( ) ✓

scanf( ) ✓

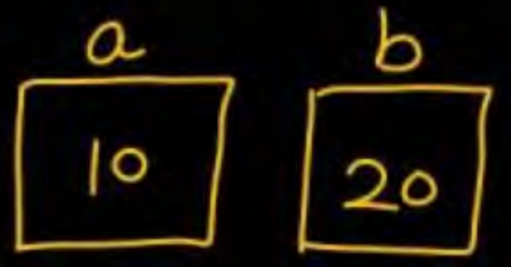
we used them

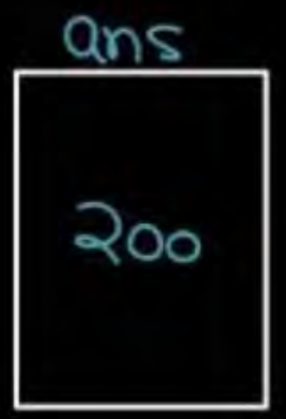
Code reusability

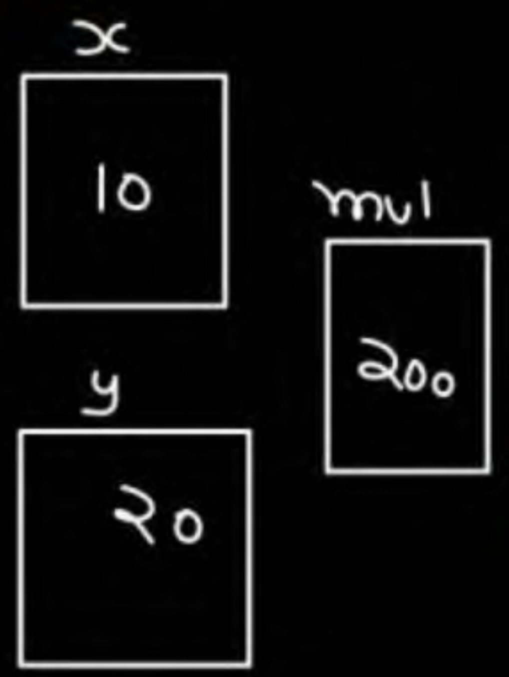
Incomplete code

```
#include<stdio.h>

void main(){

    int a=10,b=20,ans;
                  200      10 20
    ans = satishsir( a, b );
    printf("%d",ans);

}
```

```
satishsir( int x, int y )
{

    int mul;

    mul= x*y;

    return mul;

}
```



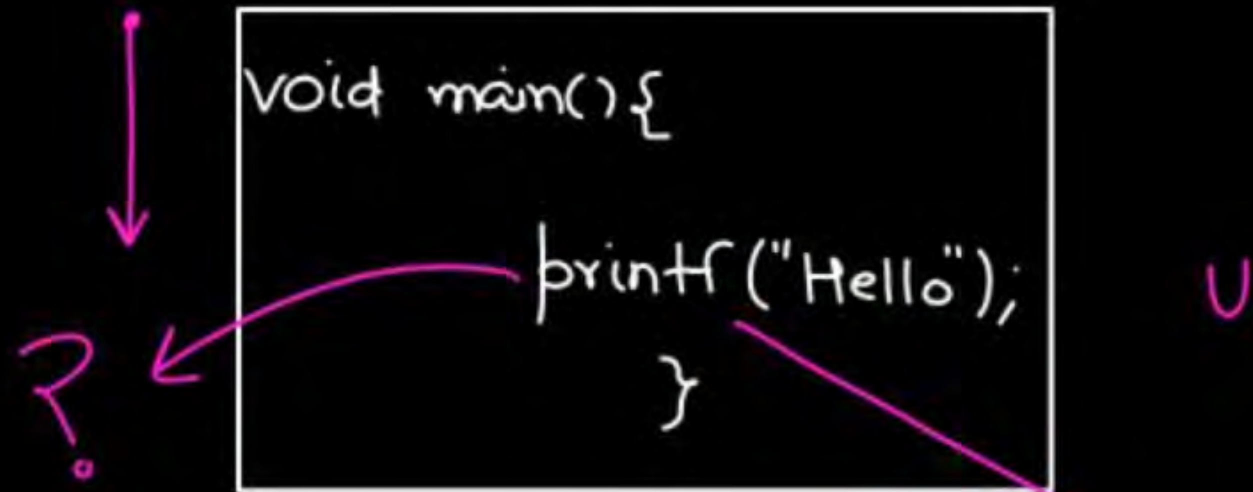




```
#include<stdio.h>
void main(){

      printf("%d",a);
         }
```

Use a without declaration

compiler info.



```
void main(){

    printf("Hello");
    }
```

using printf

Compilation

Executation

related info ⟹ header file

To avoid any C.E

```
#include<stdio.h>

void main(){

    int a = 10, b = 20, ans;

    ans = Multiply(a,b);

    printf("%d",ans);

    }
```

→ use/call

?

⇒ forward declaration

definition/body of func.

```
int Multiply(int x, int y)
{

    int res;

    res = x * y;

    return res;

}
```

```
#include<stdio.h>

int Multiply(int,int);   //forward declaration
                                prototype
void main(){

      int a=10,b=20,ans;

      ans = Multiply(a,b); //call

      printf("%d",ans);

        }
```

```
define/body
int Multiply(int x,int y)
 {
   int res;
   res = x*y;
   return res;
 }
```

```
# include<stdio.h>

   define / body
int Multiply(int x, int y)        function header
    {
     int res;
     res = x*y;
     return res;
    }
void main(){

     int a=10, b=20, ans;

     ans = Multiply(a, b);  //call

     printf("%d", ans);

        }
```

```
Short  i = 10;

Short  int  i = 10;

signed  short  int  i = 10;

signed  short  i = 10;
```

by default

All are same

```
#include<stdio.h>
      mul(int ,int);
by default void main(){
int
        int a=10,b=20,ans;
        ans = mul(a,b);
        printf("%d",ans);

    }
int mul(int x, int y){

        return x*y;

    }
```

the return type of mul function is int

Happy

```
#include<stdio.h>

void main(){
        int x;
        x = fun(10);
        printf("%d",x);
    }
double fun(int y){
    double temp=12.0;
    return temp*y;
        }
```

info save

$\boxed{\text{return type of fun function is int}}$

double $\rightarrow$ Mismatch

Error

implicit

Compilation ↓

Top to bottom

main के कोई लेना देना नहीं है

Execution

$\hookrightarrow$ main ✓

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

        int a=10,b=20,ans;

        ans = mul(a,b);

        printf("%d",ans);

              }
int mul(int x, int y){

          return x*y;

              }
```

info save

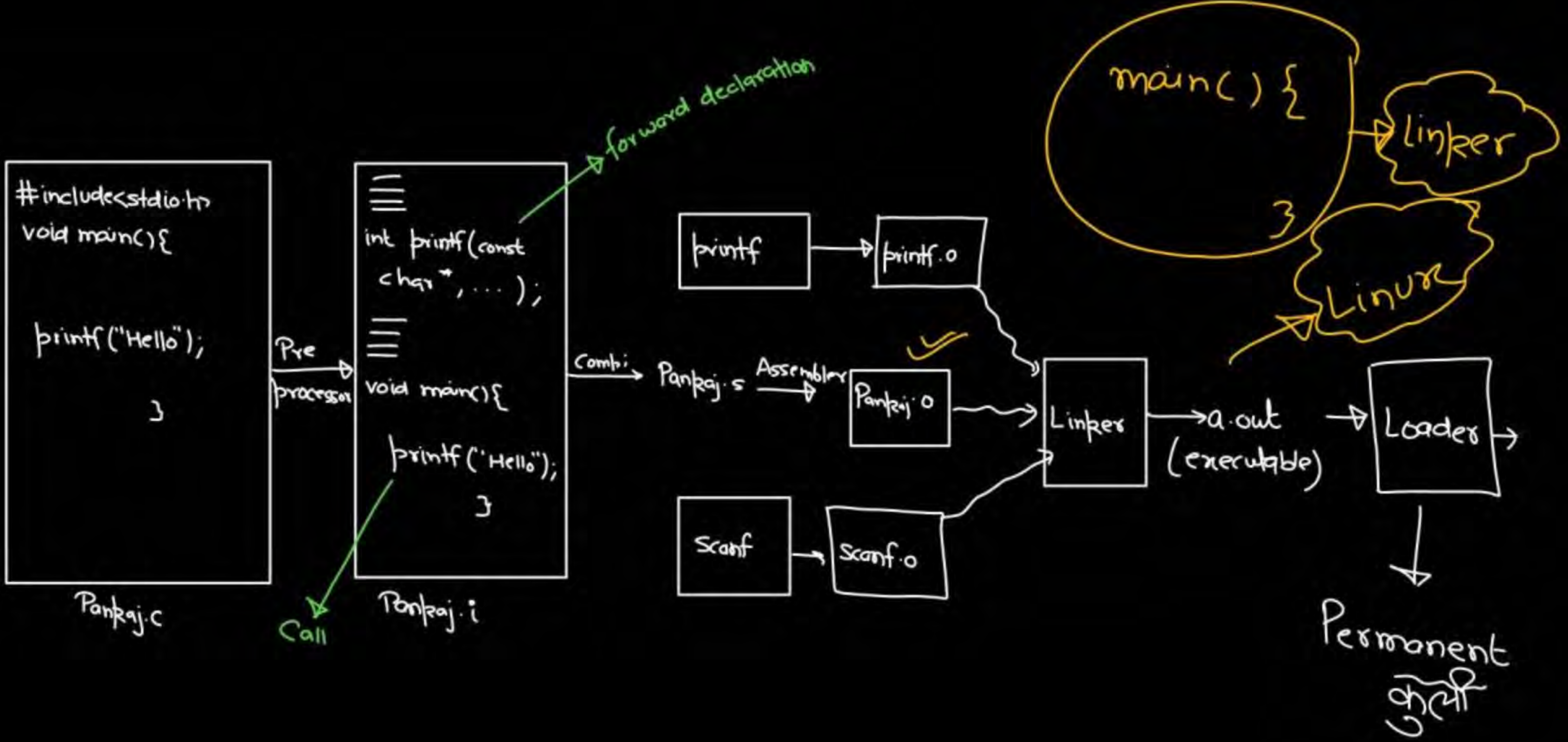
return type of mul function is int

int → same (happy)

forward declaration

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

    printf("Hello"); //call

}
```

Compile ✓



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

  printf("Hello");

        }
```

Pankaj.c

```
int printf(const
 char*, ... );

void main(){
  printf("Hello");
        }
```

Pankaj.i

```
#include<stdio.h>

int mul(int x, int y)
   {

      return x*y;

   }
```

Compile ✓

Execute ✗

Linking Error

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

    printf("Pankaj");

    }
```

use ✗

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

    int i ;
    i = printf("Pankaj");
    printf("%d", i);
    }
```

use ✓

pf → return a value

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

      int  a=10,b=20;

      mul(a,b);

          }
```

```
int mul(int a, int b)
  {
    int temp;
    temp=a*b;


   }
```

Not returning anything

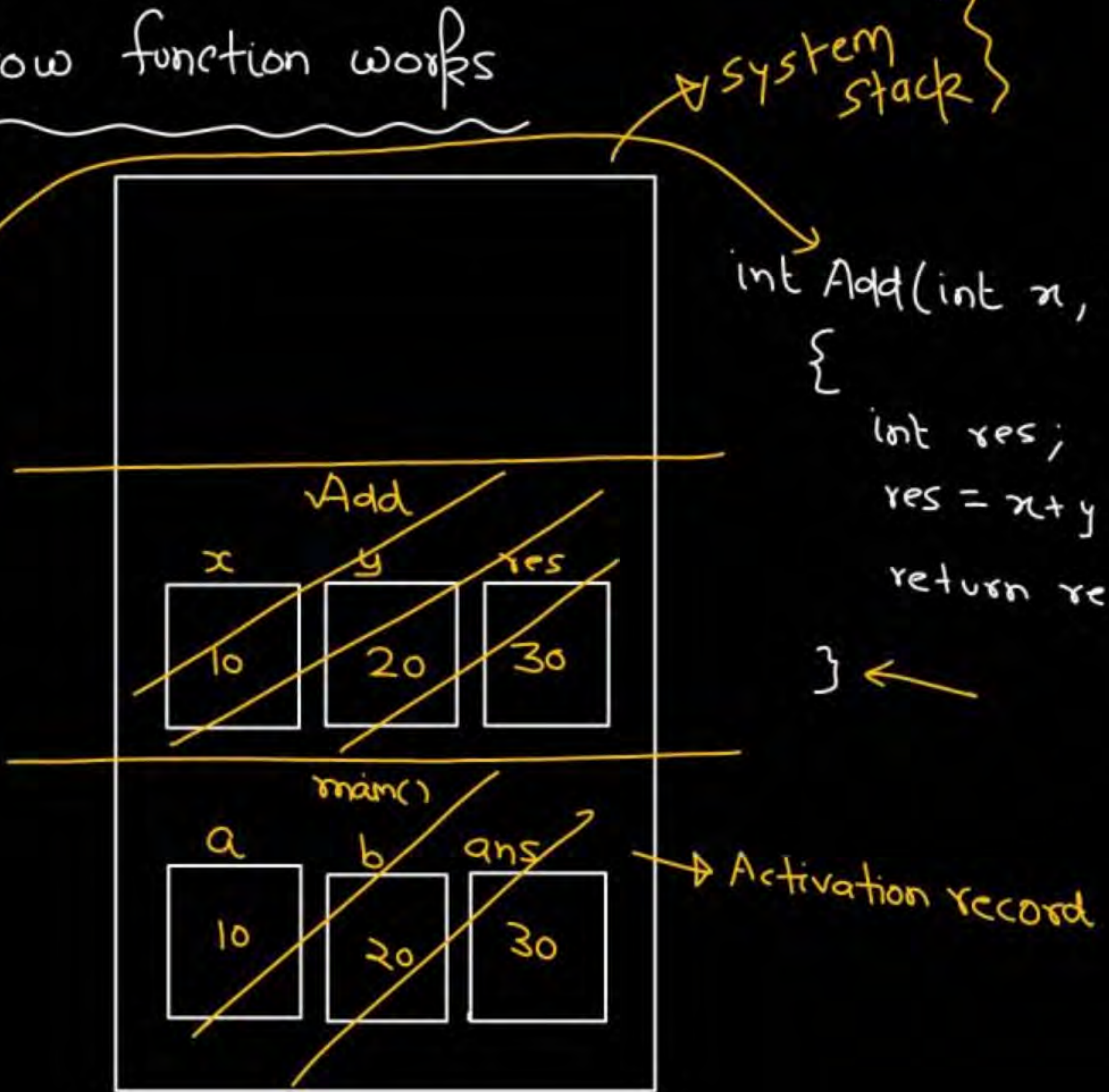
```
#include<stdio.h>
void mul(int,int);

void main(){
        int a=10,b=20;
        mul(a,b);
        }
```

```
void mul(int x, int y)
  {
      int temp;
      temp = x+y;
      printf("%d", temp);
  }
```

How function works

```
#include<stdio.h>
int Add(int,int);
void main(){
    int a=10,b=20,ans;
    ans = Add(a,b);
    printf("%d",ans);
}
```

```
int Add(int x, int y)
{
    int res;
    res = x+y;
    return res;
}
```

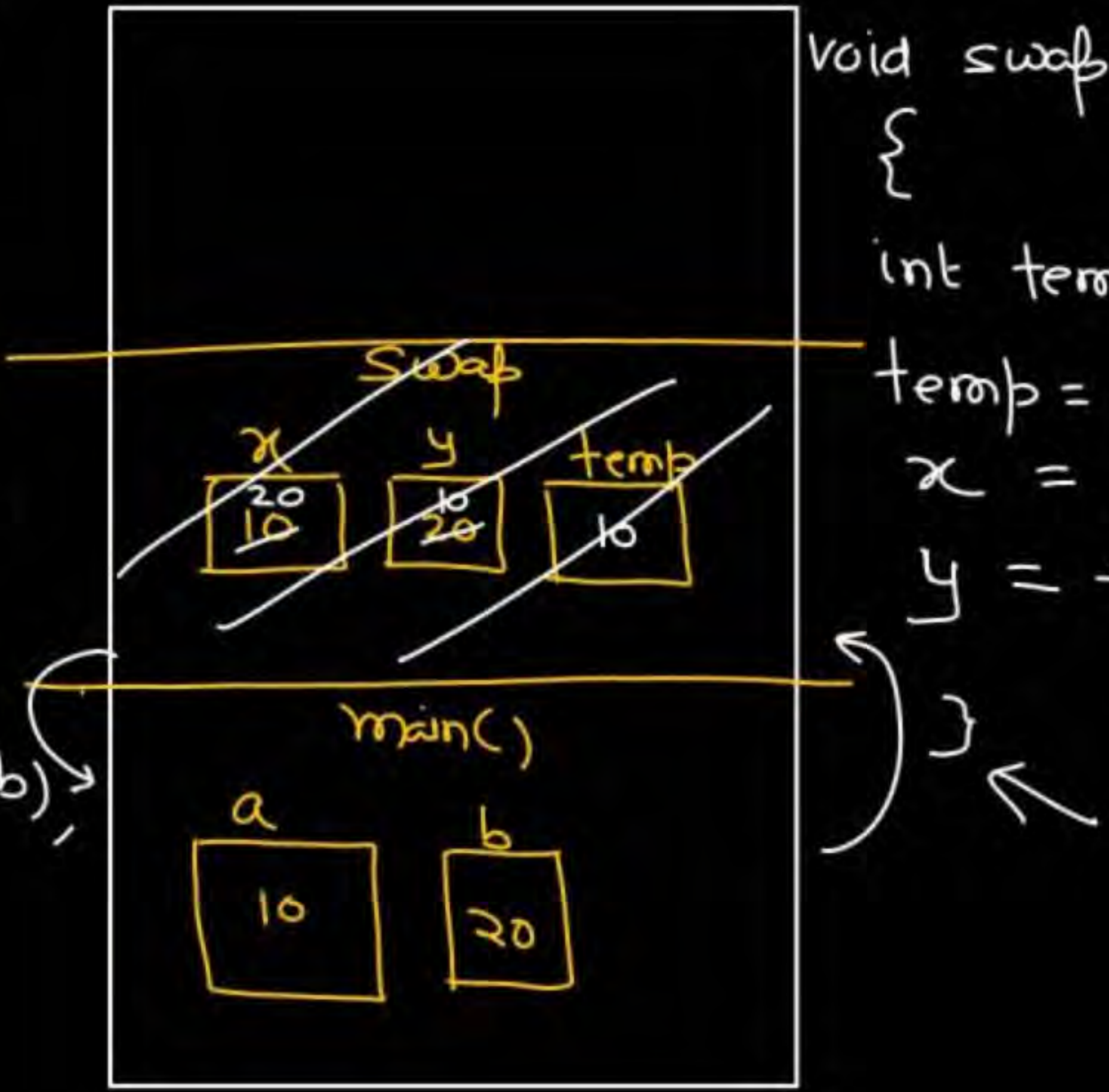


COPY

```
#include<stdio.h>

void swap(int,int);

void main(){
    int a=10,b=20;
    printf("a=%d b=%d",a,b);
    swap(a,b);
    printf("a=%d and b=%d",a,b);
}
```

10 20 ✓

```
void swap(int x,int y)
{
  int temp;
  temp = x;
  x = y;
  y = temp;
}
```



Calling

```
#include<stdio.h>

void swap(int,int);

void main(){
    int a=10,b=20;
    printf("a= %d b= %d", a, b);
    swap(a, b);
    printf("a= %d and b= %d",a,b)

}
```

10 20 ✓

$\overset{10}{a}$, $\overset{20}{b}$ $\Longrightarrow$ actual values (Actual Parameters)

Called

formal parameters

```
void swap(int x,int y)
{
int temp;
temp = x;
 x = y;
 y = temp;

}
```

main $\longrightarrow$ swap

THANK YOU GW
SOLDIERS !