CS & IT ENGINEERING

Programming in C

Functions & Storage Classes

Lec- 01

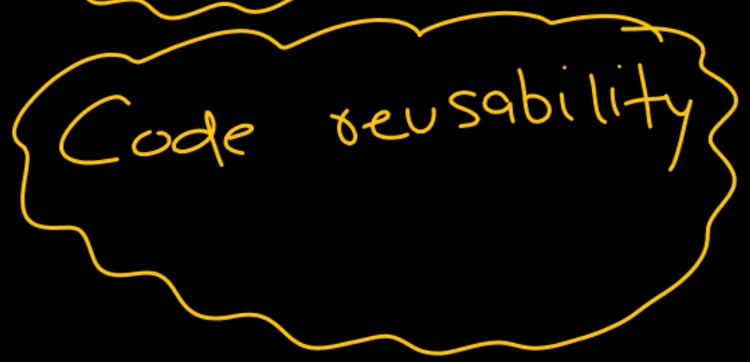
By- Pankaj Sharma sir

Functions-1

functions

printf() ✓

scanf() ✓



Incomplete code

```
#include<stdio.h>

void main(){

    int a=10,b=20,ans;

    ans = satishsir( a,b );

    printf("%d",ans);

}
```

a: 10

b: 20

ans: 200

```
satishsir( int x, int y )
{
    int mul;
    mul= x*y;
    return mul;
}
```

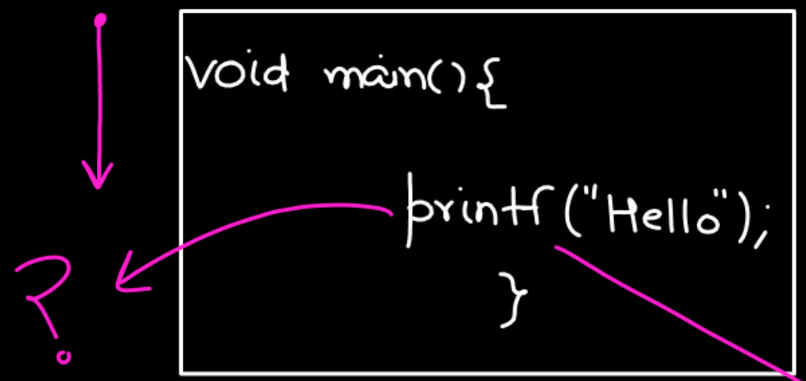
x: 10

y: 20

mul: 200

```c
#include<stdio.h>
void main(){
    printf("%d",a);
}
```

use a without declaration → compiler info.



using printf

Compilation

Execution

related info ⇒ header file

```
#include<stdio.h>

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

use/call

?

To avoid any C.E

⇒ forward declaration

definition/body of func.

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

```c
#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

```c
int Multiply(int x,int y)
{
  int res;
  res = x*y;
  return res;
}
```

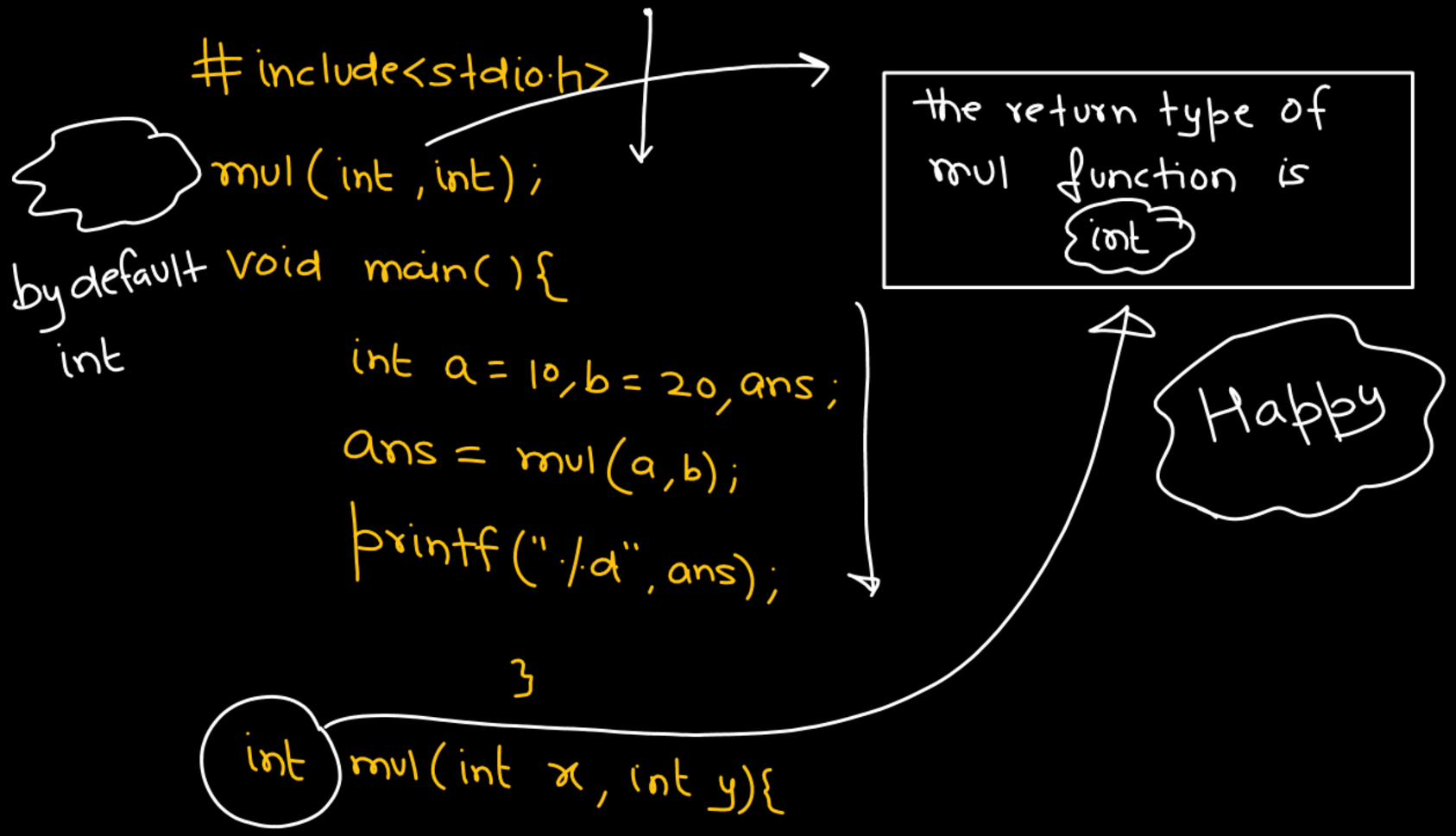
```c
#include<stdio.h>

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

      int a=10,b=20,ans;

      ans = Multiply(a,b); //call
      printf("%d",ans);
            }
```

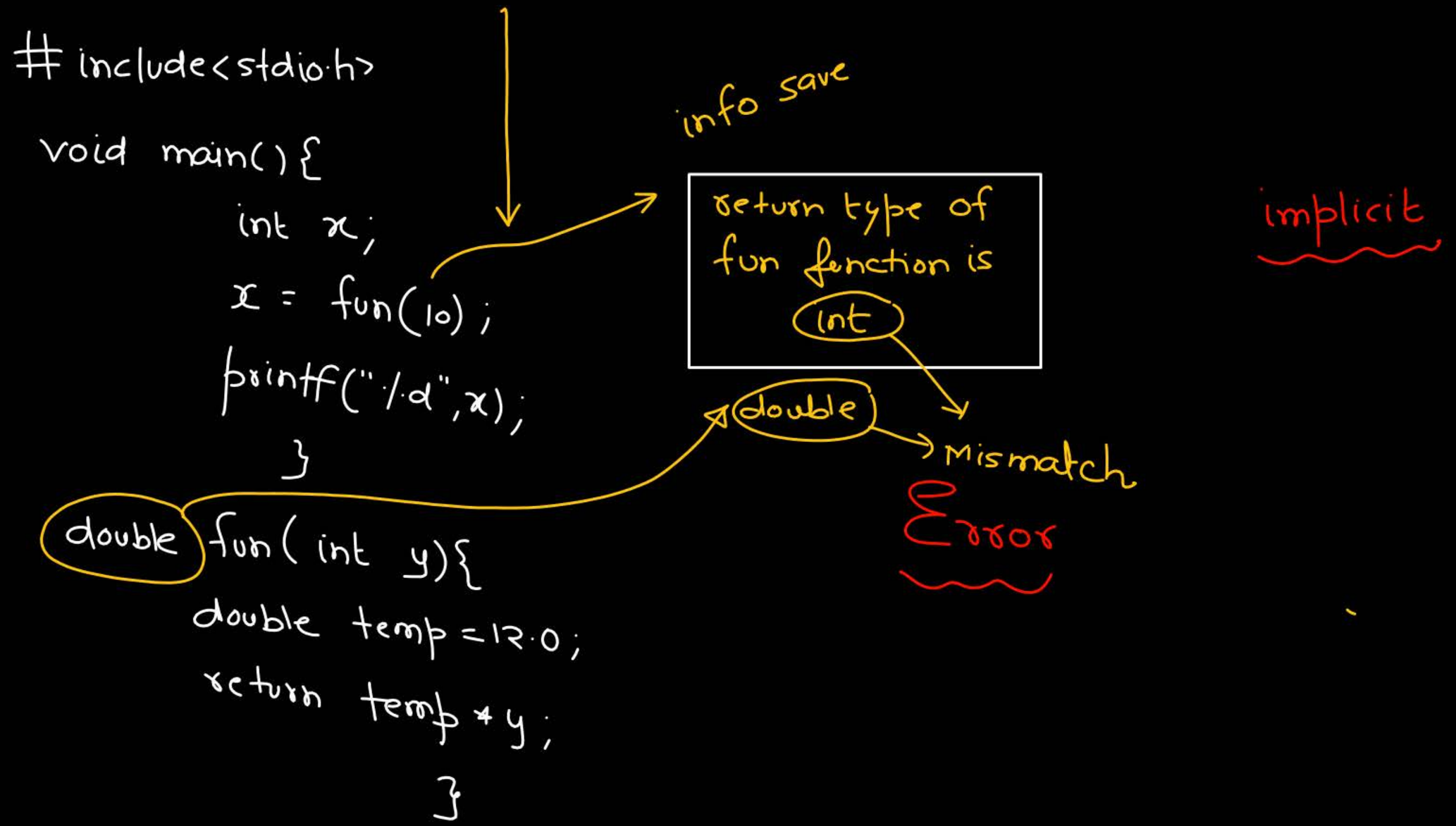
function header

```
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 int
void main(){
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
return type of fun function is
int
double
Mismatch
Error
implicit

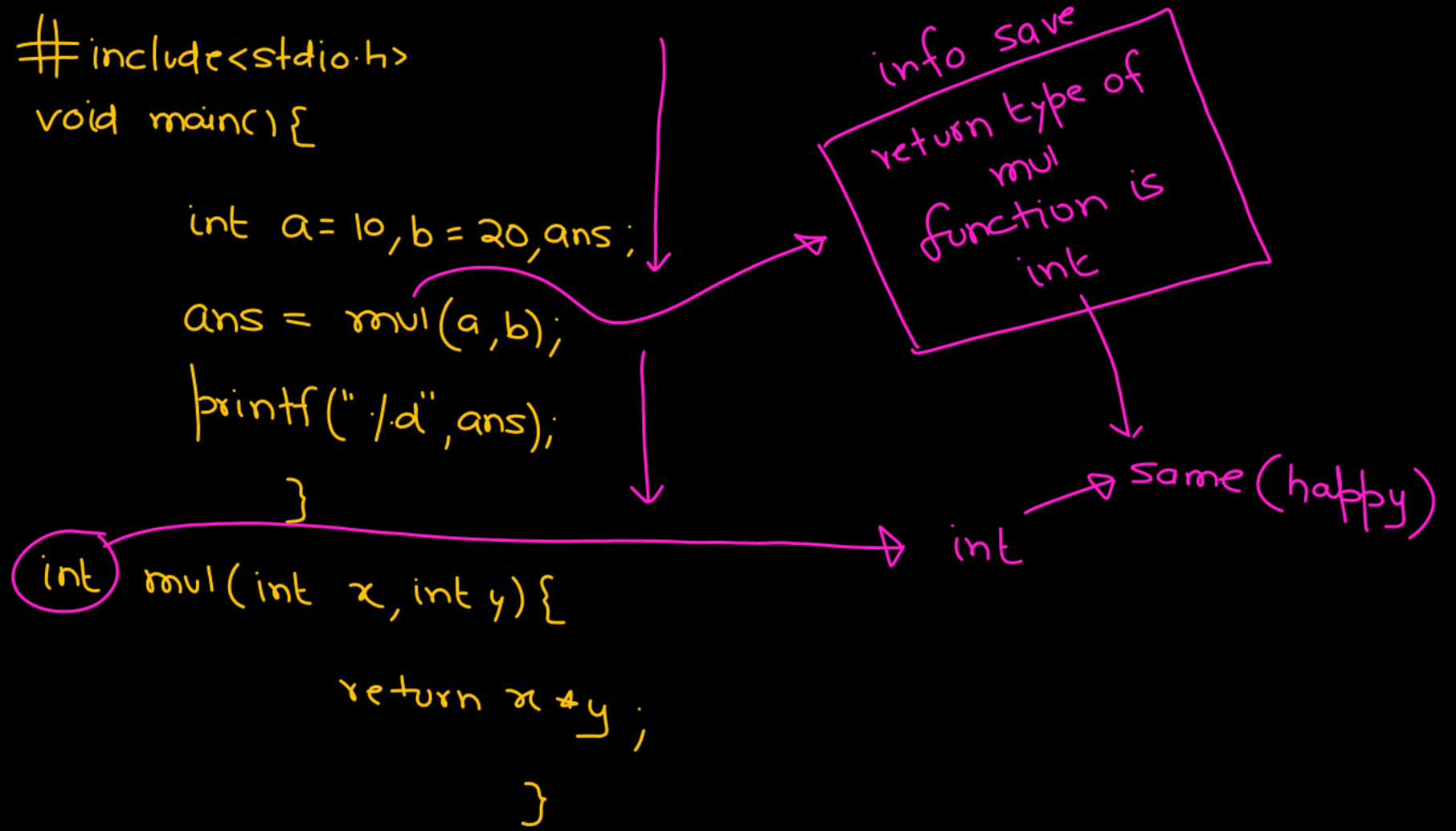
Compilation

Top to bottom

Execution

↳ main

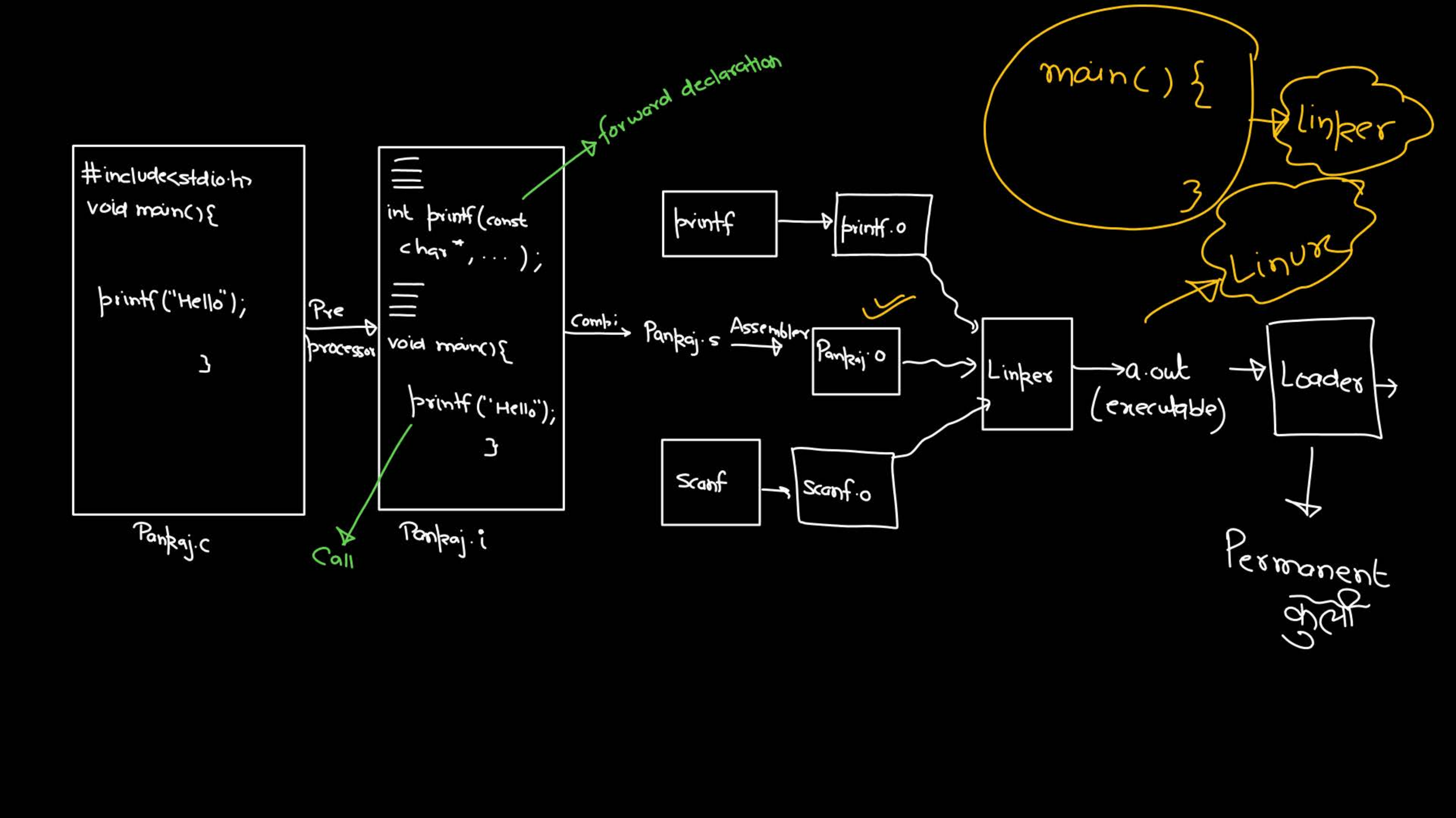
main से कोई लेना देना नहीं है


#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)

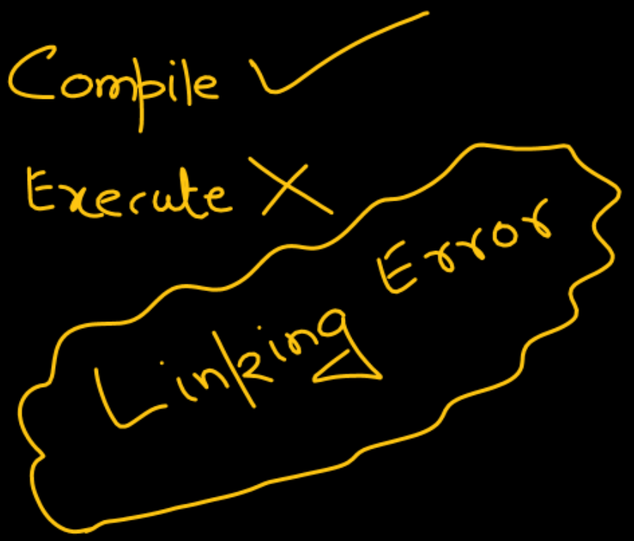
forward declaration

```
#include<stdio.h>
void main(){
printf("Hello"); //call
}
```

Compile ✓


#include<stdio.h>
void main(){
printf("Hello");
}
Pankaj.c
Pre
processor
int printf(const
char*, ...);
void main(){
printf("Hello");
}
Pankaj.i
forward declaration
Call
Combi
Pankaj.s
Assembler
printf
printf.o
Pankaj.o
Scanf
Scanf.o
Linker
a.out
(executable)
Loader
main(){
3
linker
Linux
Permanent
कुछ भी

```c
#include<stdio.h>

int mul(int x, int y)
  {
    return x*y;
  }
```



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

    printf("Pankaj");

}
```

use ✗

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

use ✓

pf → return a value

```c
#include<stdio.h>
void main(){
    int a=10,b=20;
    mul(a,b);
}
```

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

Not returning anything

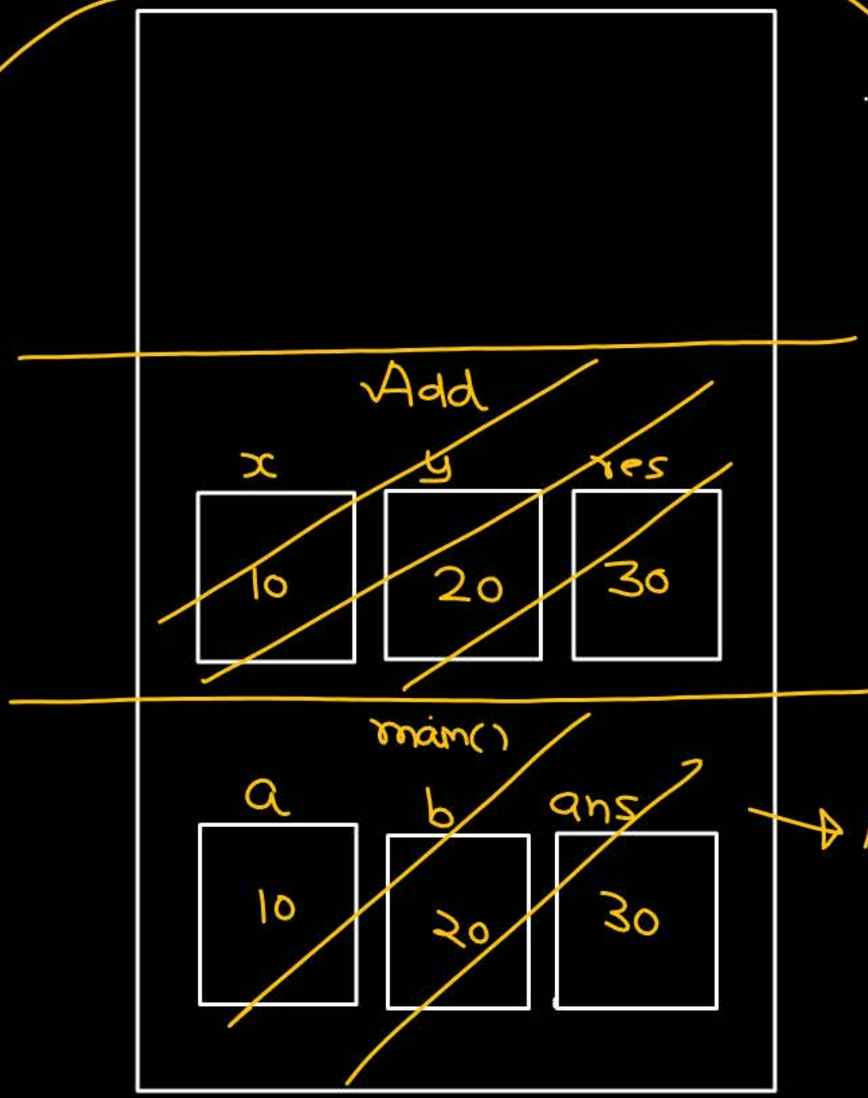
```
#include<stdio.h>
void mul(int,int);

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);
}
```

10 20



system stack

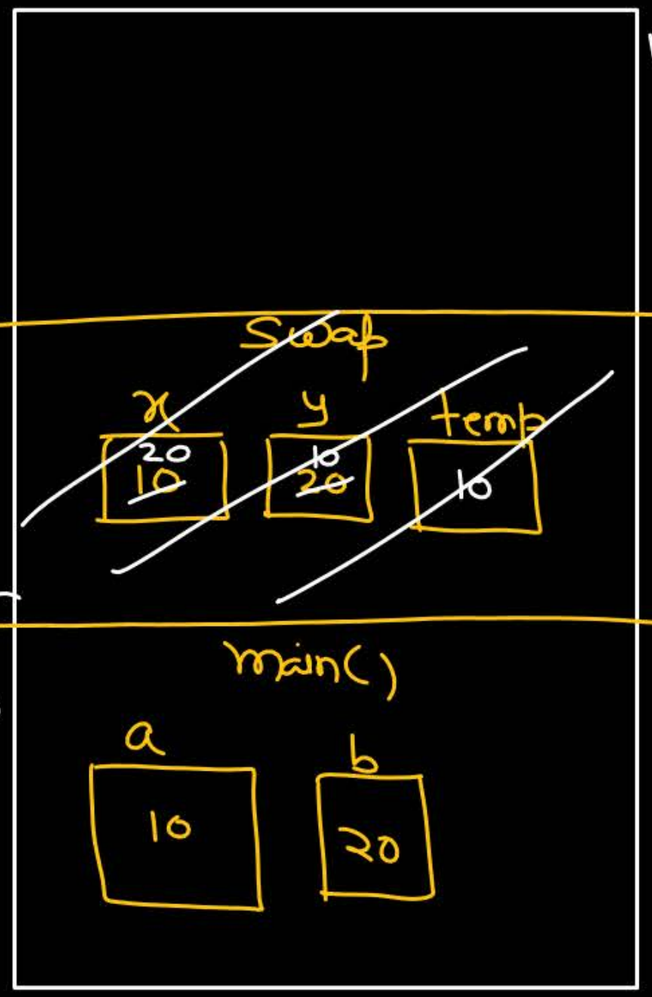
Activation record

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

```c
#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

Copy




```c
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 and b=%d",a,b);
}
```

Calling

10 20

10 20 ⟹ actual values (Actual Parameters)

Called

formal parameters

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

main ⟶ swap


THANK YOU GW
SOLDIERS !