CS & IT ENGINEERING

Programming in C

Functions and Storage Classes
Lec- 01

By- Pankaj Sharma sir

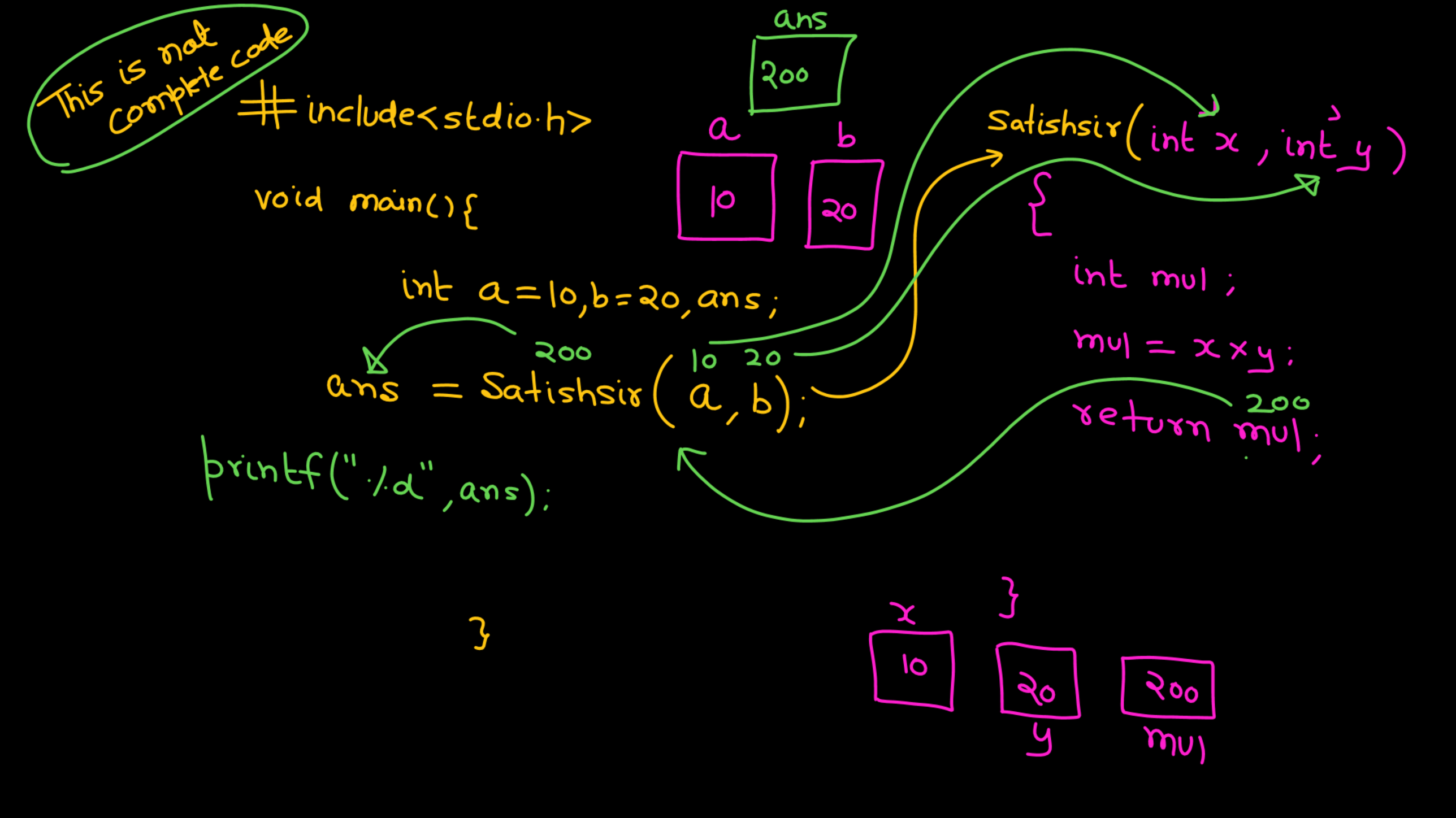
Functions

Function

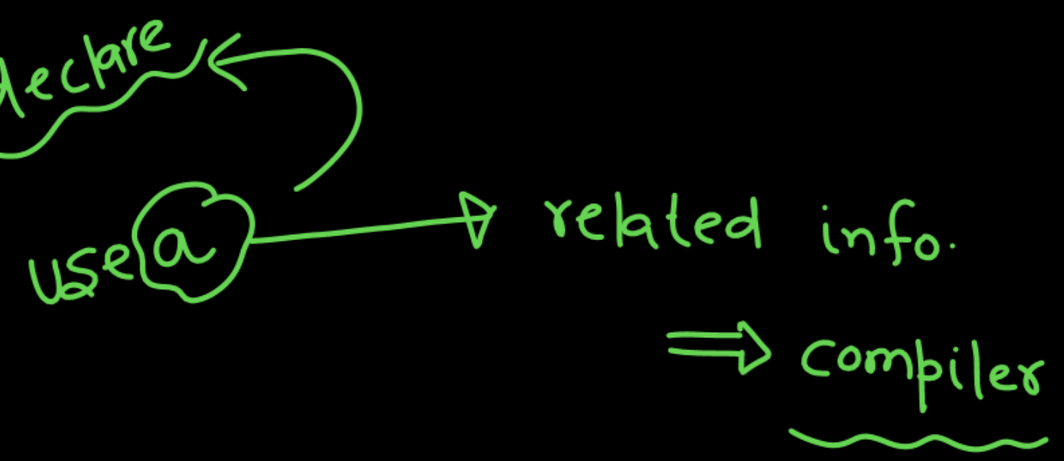
printf ✓

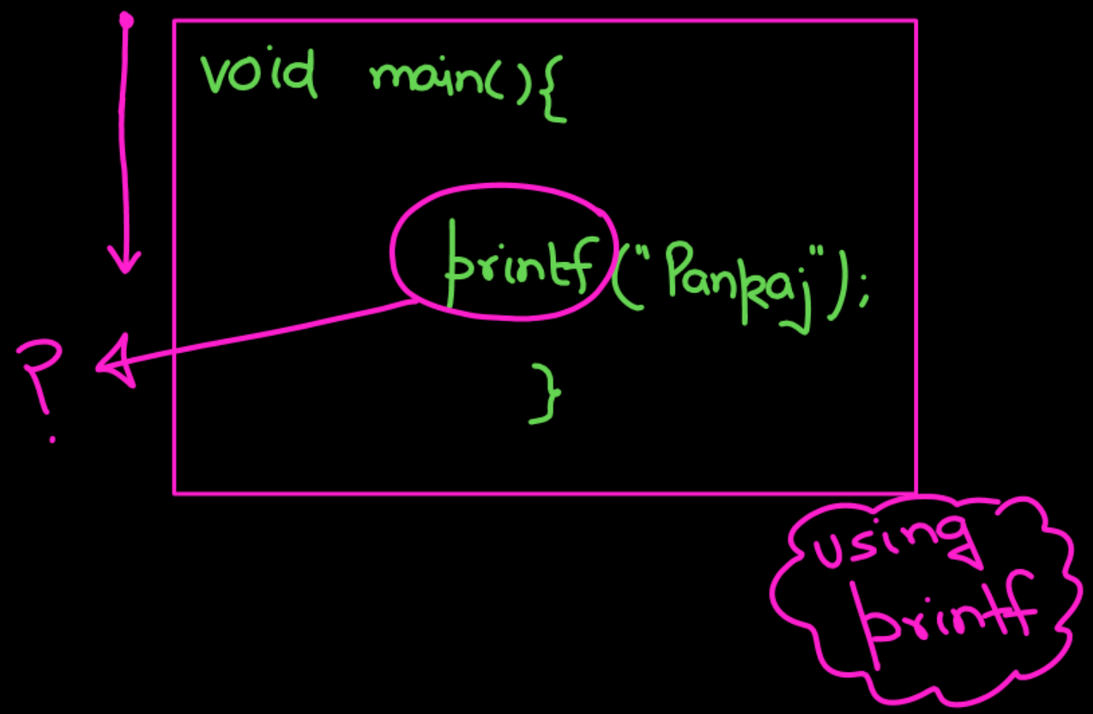
scanf ✓

Code - reusability


This is not complete code
#include<stdio.h>
void main(){
int a=10,b=20,ans;
ans = Satishsir(a,b);
printf("%d",ans);
}
ans
200
a
10
b
20
Satishsir(int x, int y)
{
int mul;
mul = x*y;
return mul;
200
10 20
x
10
y
20
mul
200

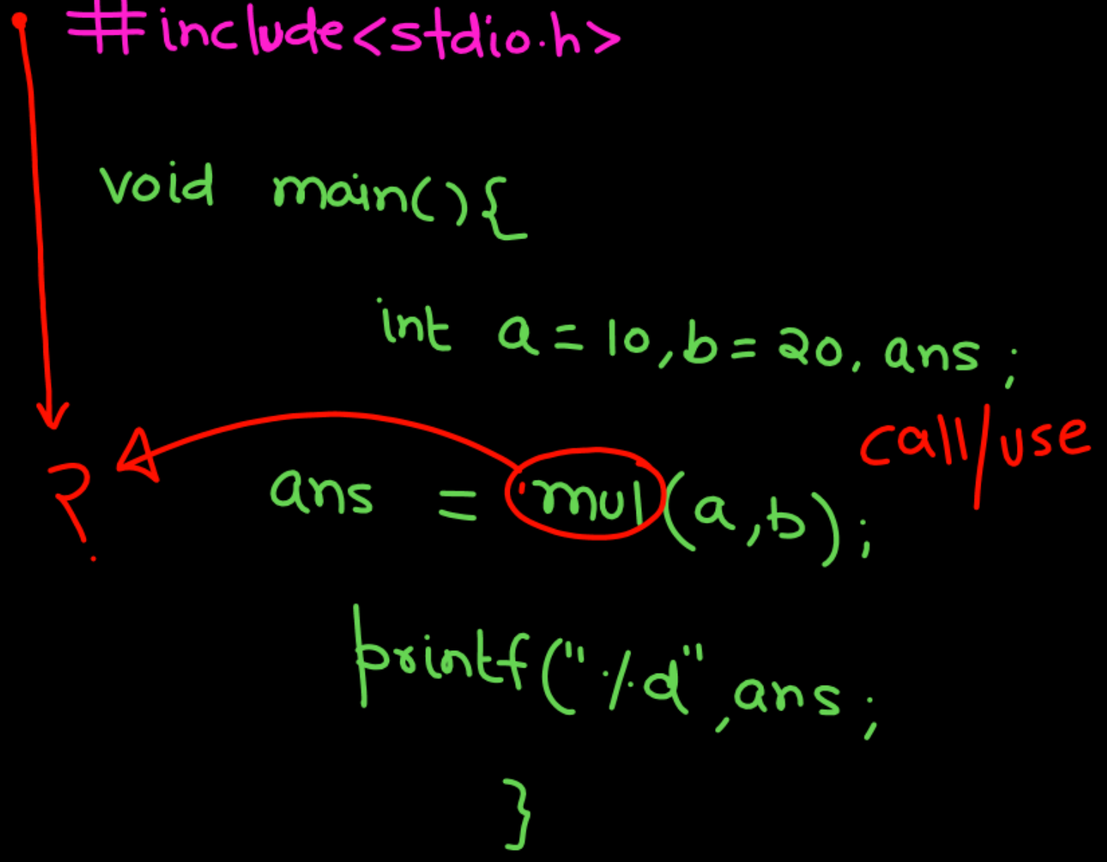
```
#include<stdio.h>
void main(){

        printf("%d",a);

}
```

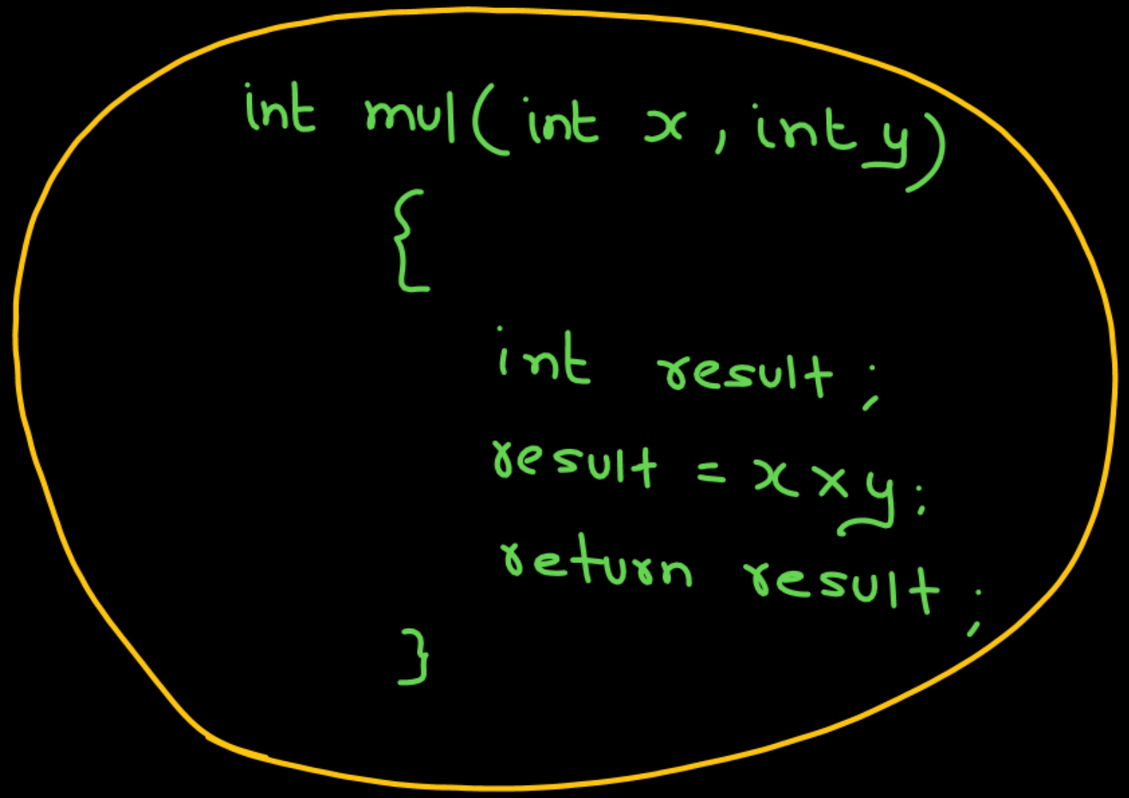




Compilation :→

Execution





body/defination

To avoid compilation Error

→ forward declaration

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

Prototype
forward decleration

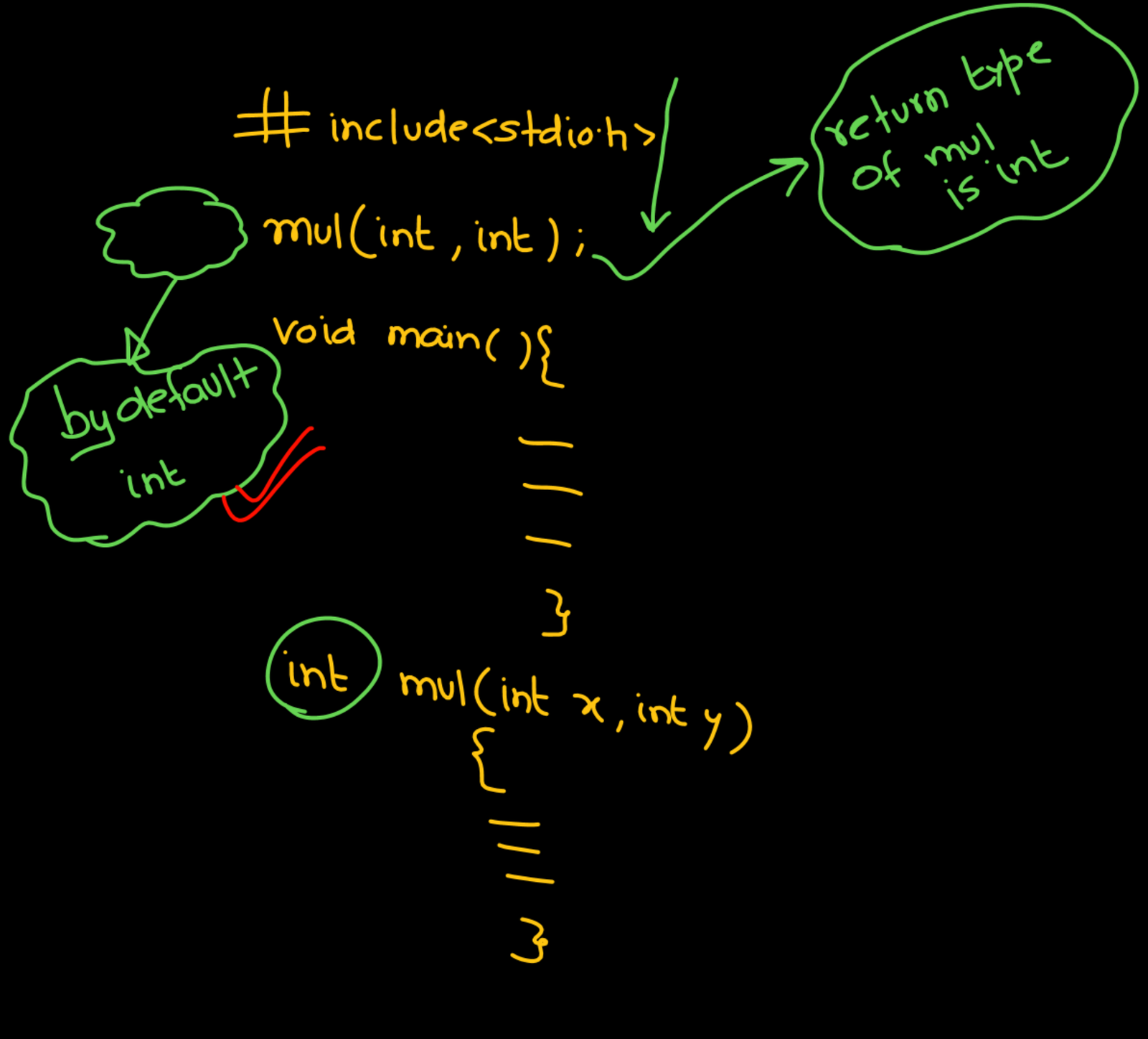
→ call/use

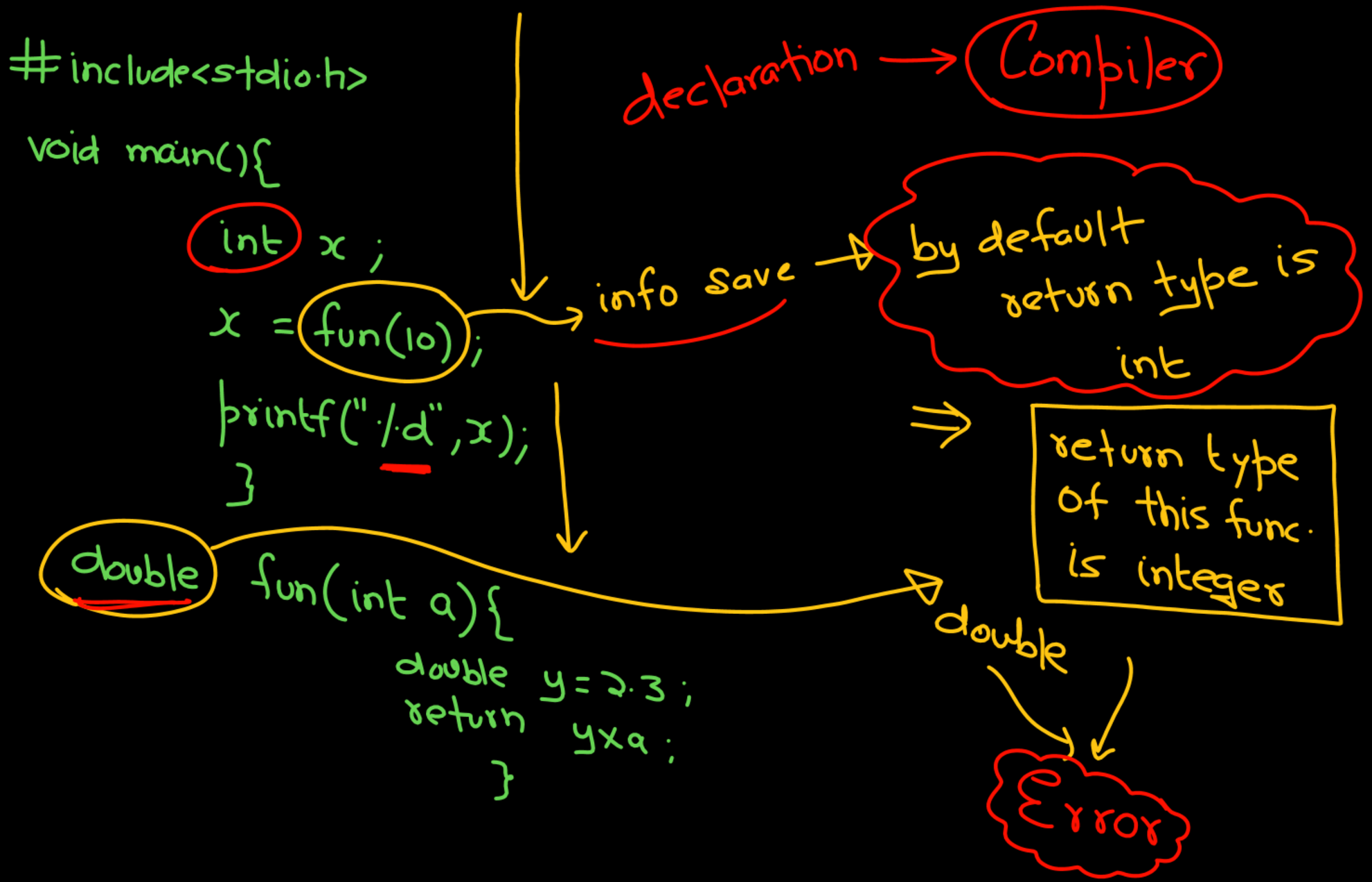
body/defination

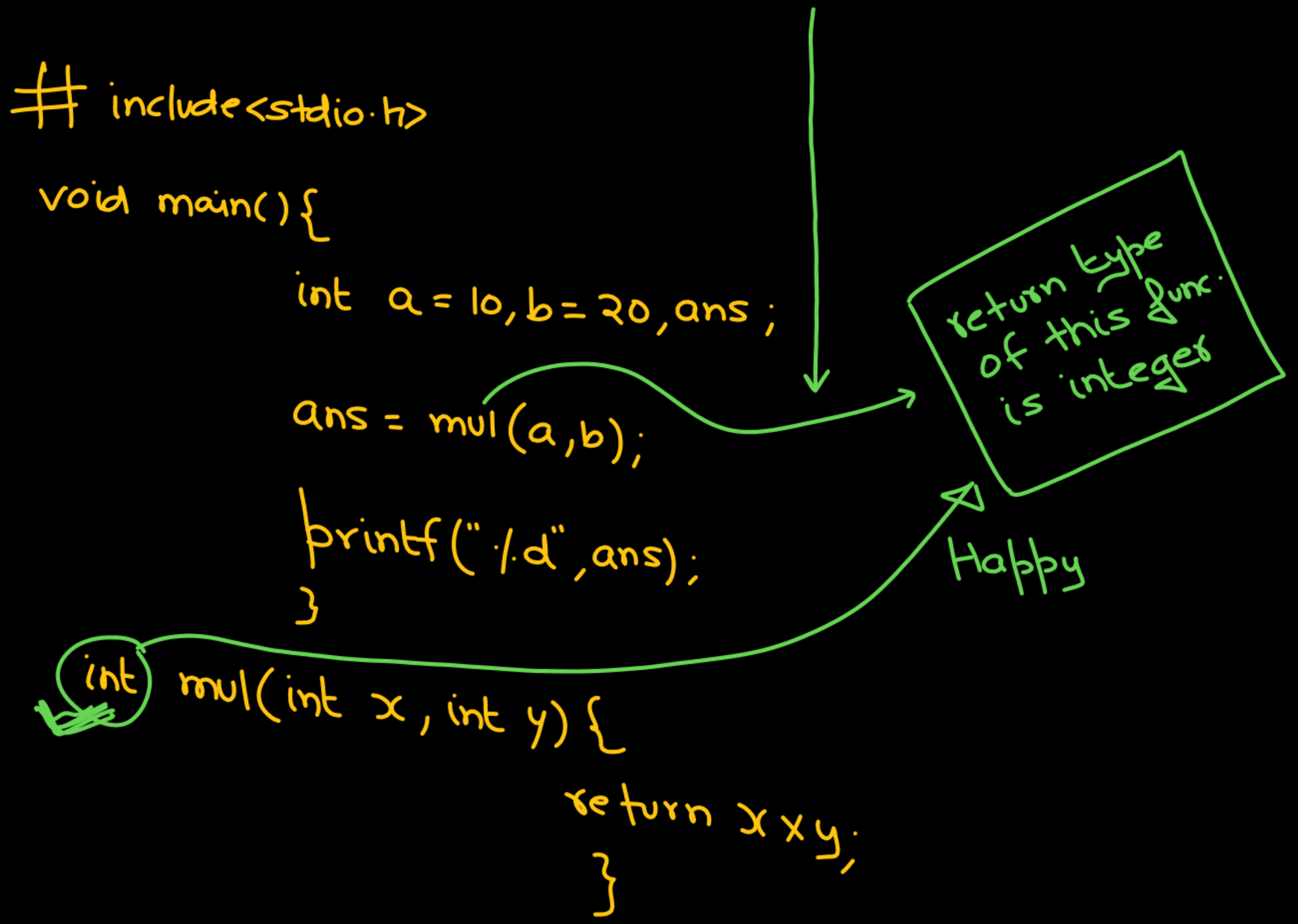
```
int mul(int x,int y)
{
    int result;
    result = x*y;
    return result;
}
```

function header

```
#include<stdio.h>
int mul(int x, int y){
    int result;
    result = x*y;
    return result;
}
void main(){
    int a=10,b=20,ans;
    ans = mul(a,b);
    printf("%d",ans);
}
```


#include<stdio.h>
mul(int, int);
return type of mul is int
by default int
void main(){
}
int
mul(int x, int y)
{
}


#include<stdio.h>
void main(){
int x ;
x = fun(10);
printf("%d",x);
}
double fun(int a){
double y=2.3;
return yxa;
}
declaration
Compiler
info save
by default return type is int
return type of this func. is integer
double
Error


#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;
}
return type of this func. is integer
Happy

forward declaration

```
#include<stdio.h>

void main(){

    printf("Pankaj");
}
```

use/call

Compile ✓

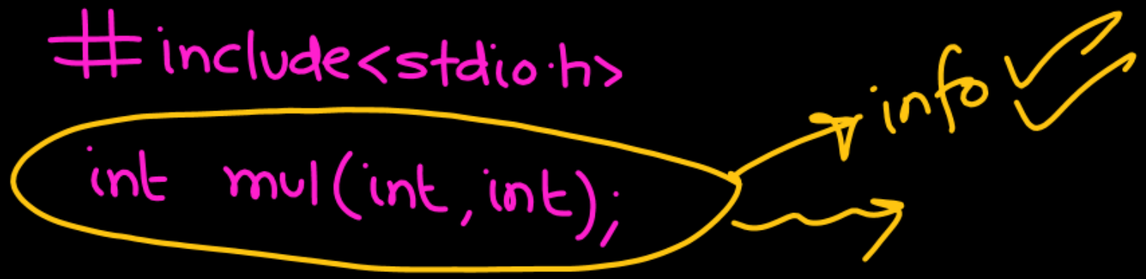
Execute ✓

forward dec.

```
#include <stdio.h>

void main(){

}
```

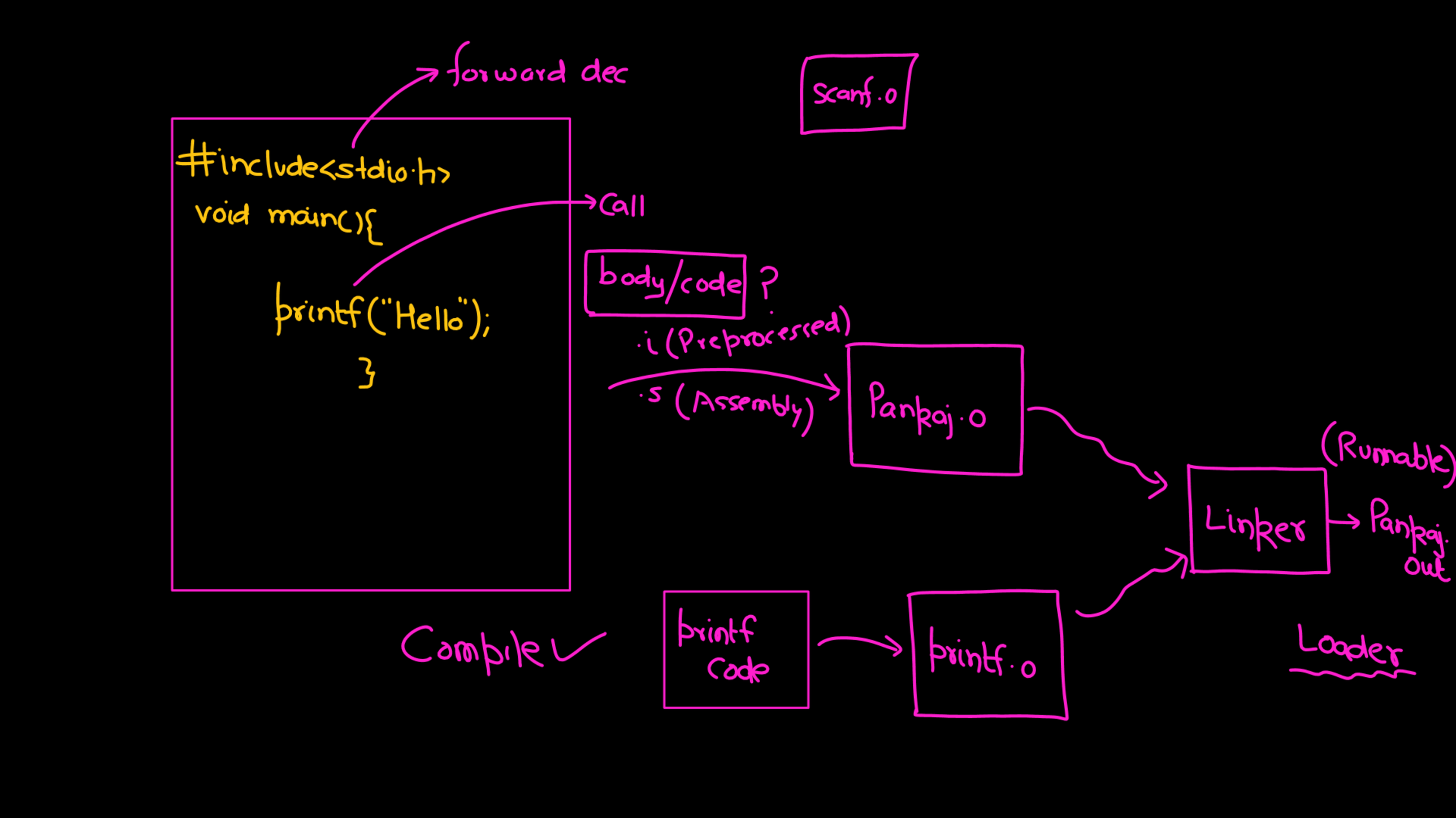
Compile ✓

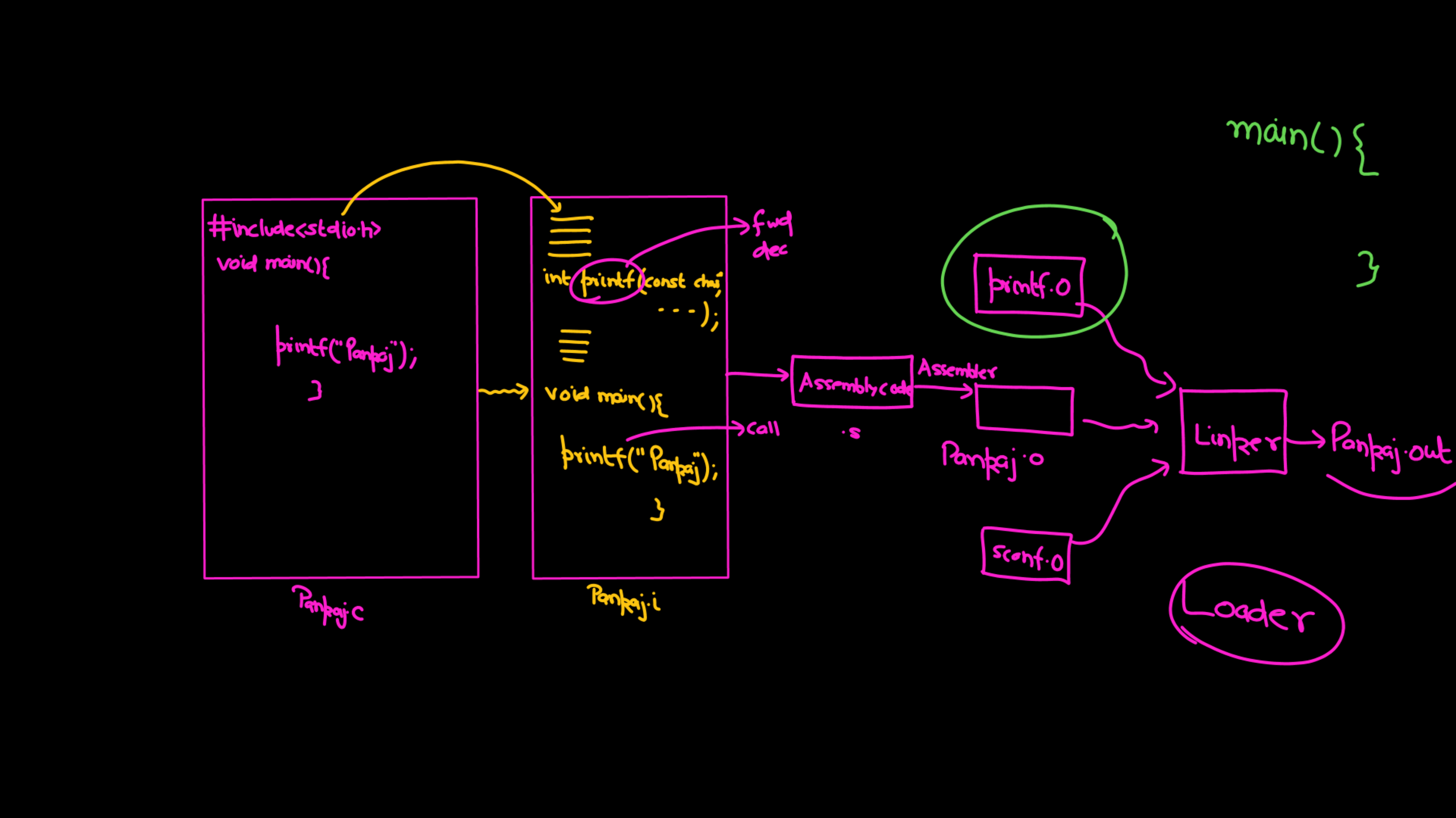
Execute ✓

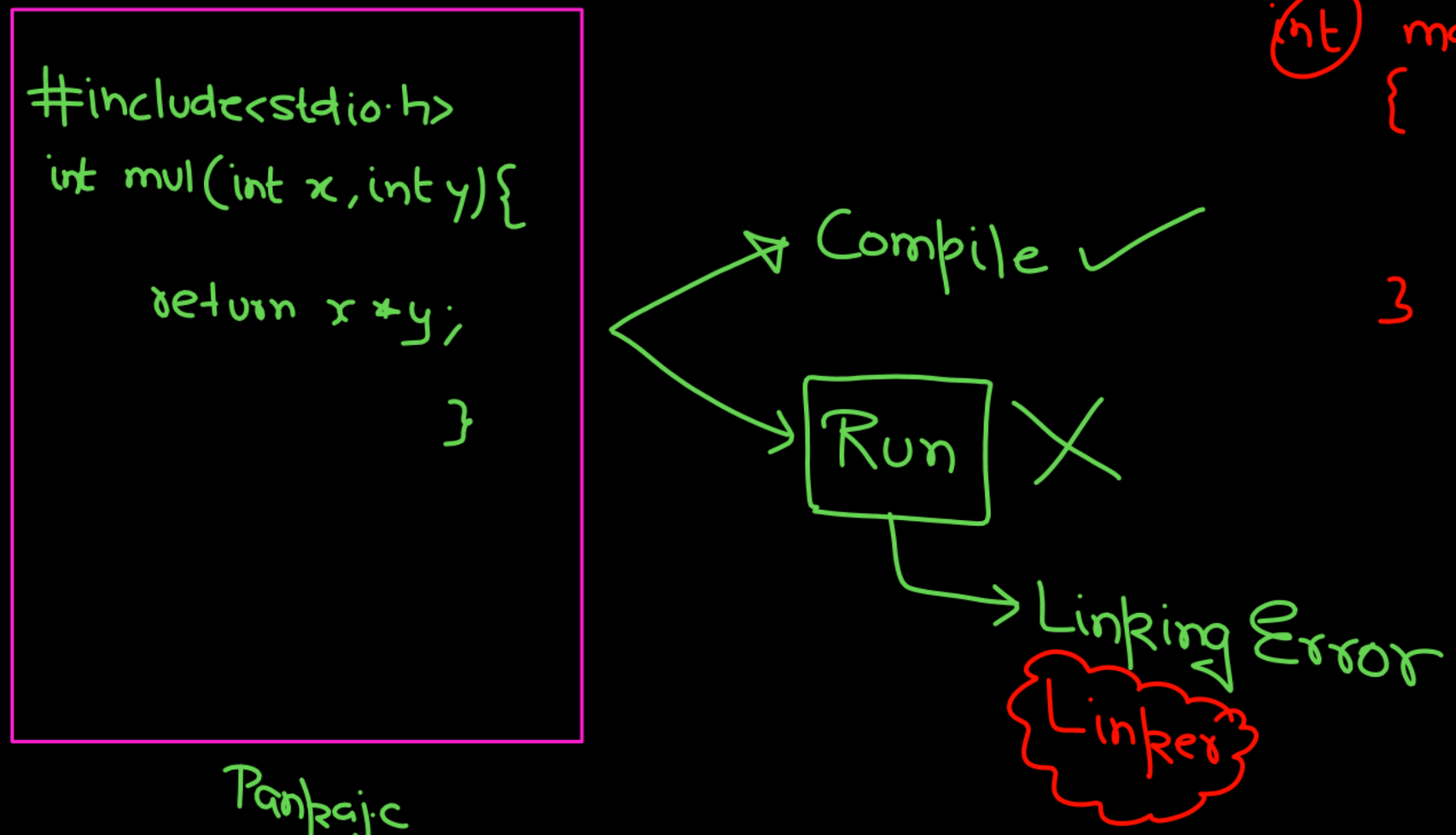



```
void main(){
    printf("Hello");
}
```

Compile


forward dec
#include<stdio.h>
void main(){
printf("Hello");
}
Call
Scanf.o
body/code ?
.i (Preprocessed)
.s (Assembly)
Pankaj.o
(Runnable)
Linker
Pankaj.out
Compile
printf Code
printf.o
Loader

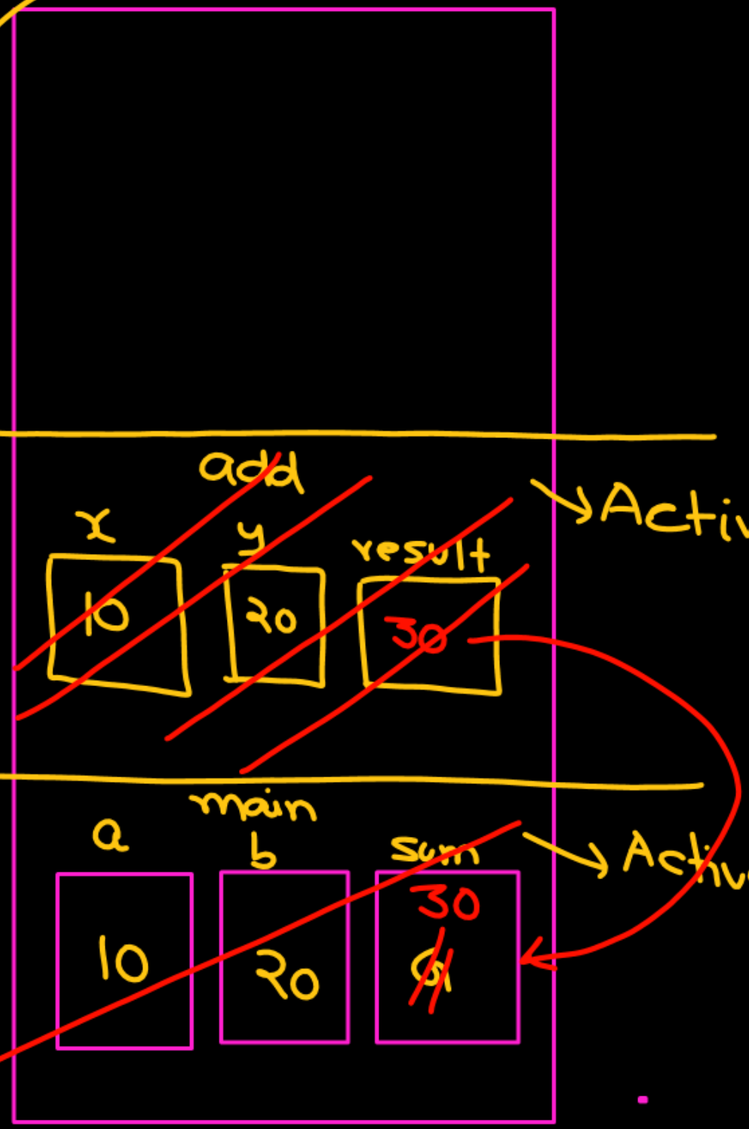

main(){
}
#include<stdio.h>
void main(){
printf("Pankaj");
}
Pankaj.c
int printf(const char*, ...);
fwd dec
void main(){
printf("Pankaj");
}
Pankaj.i
call
Assembly code
.s
Assembler
printf.o
Pankaj.o
scanf.o
Linker
Pankaj.out
Loader


#include<stdio.h>
int mul(int x, int y){
return x*y;
}
Pankaj.c
Compile
Run
Linking Error
Linker
int main()
{
}
void main()
{
}

How a function works

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

```
int add(int x,int y)
{
  int result;
  result = x+y;
  return result;
}
```



30

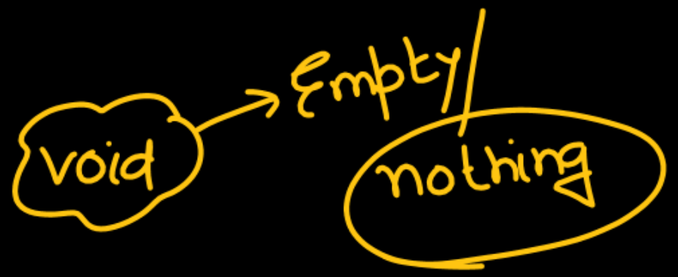
```c
#include<stdio.h>
int add(int,int);
void main(){
    int a=10,b=20,sum;
    sum=add(a,b);
    printf("%d",sum);
}
```

calling function

actual arguments

```c
int add(int x,int y)
{
    return x+y;
}
```

formal argument




```
printf("Pankaj");
```

```
int i;
i = printf("Pankaj");
```

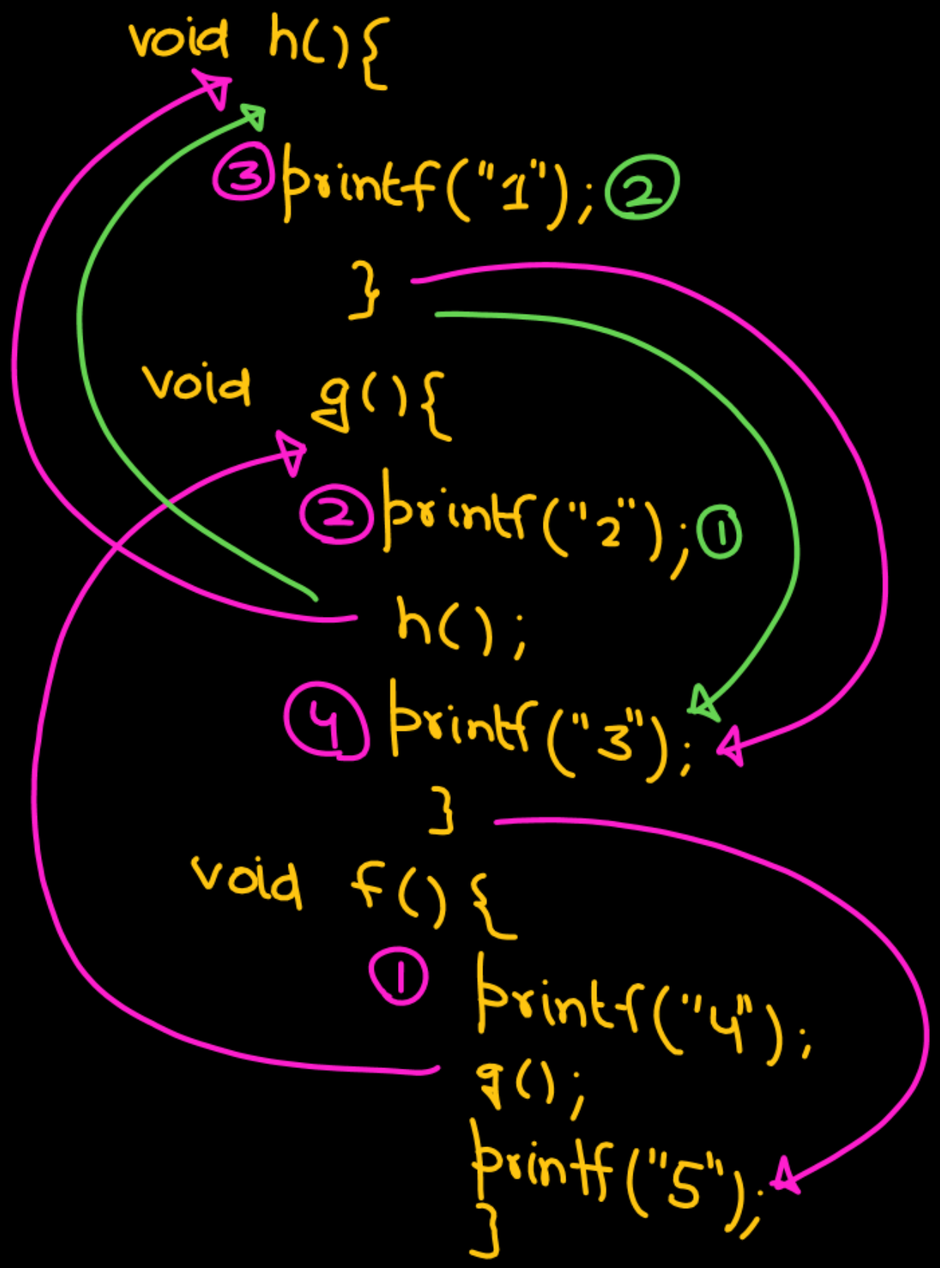
return integer

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

```c
void Add(int x, int y)
{
 int result;
 result = x+y;
 printf("%d",result);
}
```

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

printf("Pankaj");
```

```
int mul(int x,int y)
    {
       int result;
       result= x xy;

    }
```

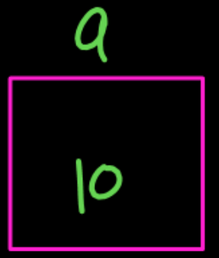


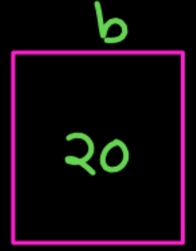

```
void main(){
  f();
  g();
  h();
  f();
}
```

o/p: 42135 213 142135

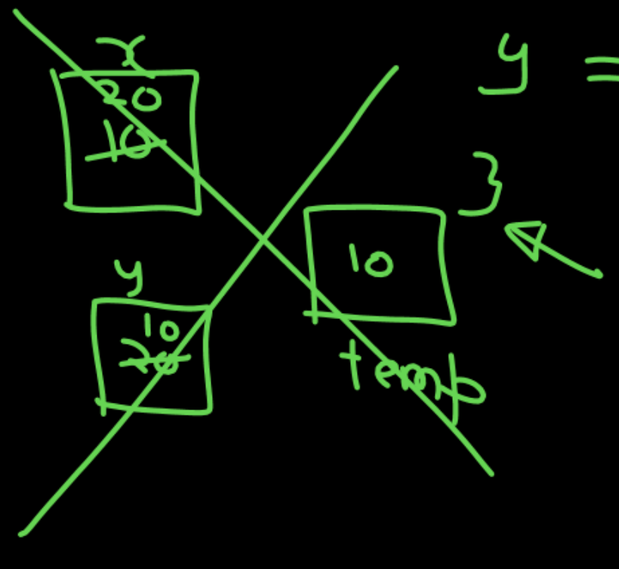
```
#include<stdio.h>
void swap(int, int);
void main(){
    int a=10,b=20;
    printf("Before swap");
    printf("a=%d,b=%d",a,b);
    swap(a,b);
    printf("a=%d,b=%d",a,b);
}
```

Before swap a=10, b=20 a=10, b=20






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




THANK YOU GW
SOLDIERS !