

L-11

factorial.??

$$n! = n \times (n-1) \times (n-2) \times \dots \times 1$$

$$5! = 5 \times 4 \times 3 \times 2 \times 1$$

$$= 1 \times 2 \times 3 \times 4 \times 5$$

Q7=

Q. Can you calculate the factorial of an int.

input  $\rightarrow$  5 :  $(5!)$   $\rightarrow$  multiply all values from 1 to 5.

Scam = 0



```
3  
( cout << fact; )
```

function. a group of program/statements / commands which are combined into a unit, given a name, and perform a specific task.

It can be called/invoked at any time or from any where in the program.

- code reusability
- better organization
- less code
- code readable

Syntax :

return type      parameters  
 ↓  
 name (-----) {

task:  
statements / comeds  
return statement;

properties:

- it has a specific name.
- it may / may not has }  
a return type
- it may take some  
parameters.

calculate factorial

→ input: (n) ?!

parameters  
arguments

→ output: (n!)

return value

① function must be ~~defined~~<sup>declare</sup> before it is called / invoked.

② scope / visibility  $\Rightarrow$  function should be visible to one who calls it.

global

global scope

$\downarrow$

outside

main

①

```
int main( ) {
```

```
    X X X
```

```
}
```

②

1 #include  
2 using namespace std

line-by-line

Q. who calls main() ??

4 int main (void) {

5  
6  
7  
8 }  
9  
10  
11

task : logic : work

main  
+ main is a special func.  
+ return 0 → no errors

1 #include

2 using namespace

3 int fact (int n);

4 int main() {

5 - - -

6 cout << fact(n);

7 }

8

9

10

int fact (int n) {

- - -

- - -

- - -

}

declare

defining a variable can this work

(int n);

Simply declare the fact() here  
and define it later

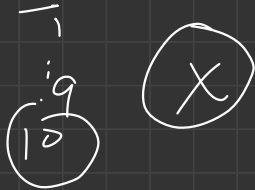
Should the compiler check  
for a function named fact() ??

Compiled

C/C++

- entire code is  
read (error check)  
and translated to  
machine code.

- then it is executed  
all at once



Interpreted

Python / JavaScript

- read line-by-line

- read a line

✓ error check the line

✓ execute the line

9.

(10)



function ————— can be written / declared-defined  
before main

or

the signature is mentioned before main  
≠  
defined later

H.W.

Q. Print all prime numbers between 2 and n.

main function



variables  
↙

( - - )  
factorial



variables  
↙

print fact-



variable  
↙

local variables : variables defined within a function or parameter  
variables are local to a function.

i.e. they are specific to the function

— are unique for that function

— { exist only while the func. is being executed }

int n;

— created <sup>automatically</sup> when the function is called  
Deleted when function completes  
by the function itself.

→ call → execution → completion

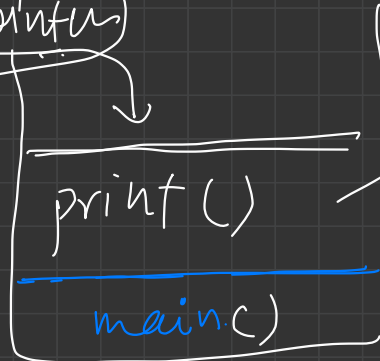
```
int fact(int n) {  
    - int, t  
    - return f;  
}
```

main() {

18     main()

22     print()

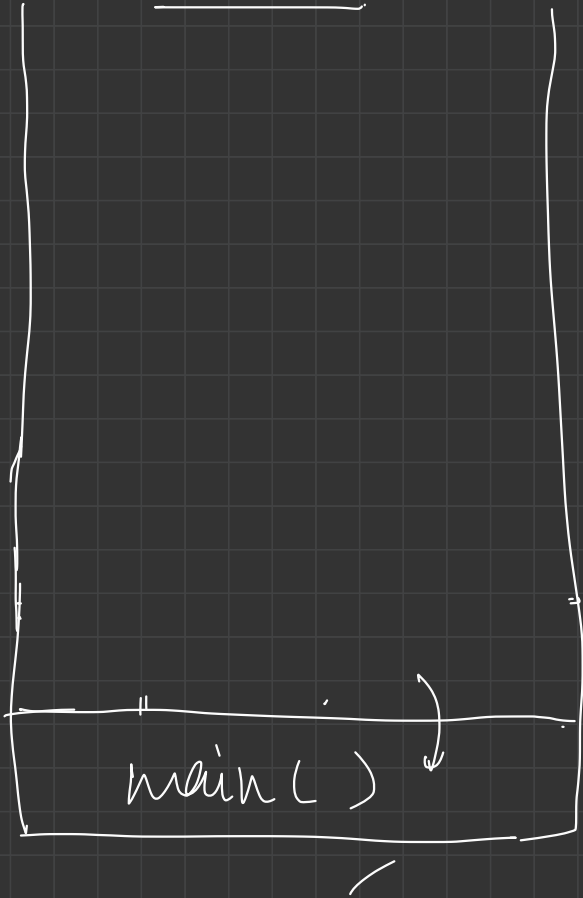
Stack  
function pointer



int n;

call, execution, comp.

call stack : topmost function is currently being executed.



18. main()

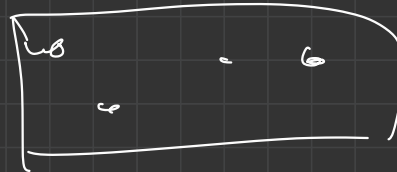
22. print()

23. print(b)

24.

25.

26.



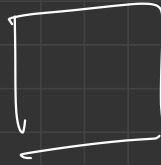
Call Stack

logical structure

memory  $\Rightarrow$  RAM



Call stack x x



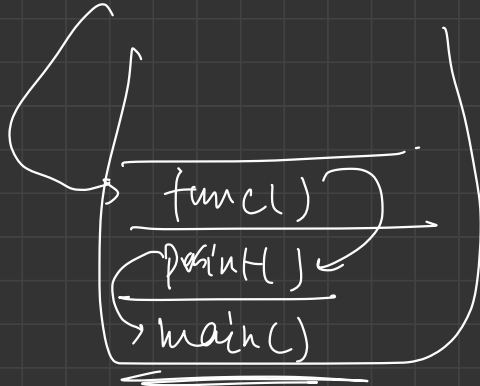
logic  $\Rightarrow$  control flow



call stack

it is the way in which functions work.

$$(2 \times 3) - (5 \times 1) - 4(1 \times 2(-3 \times 2 + 5))$$



$\Rightarrow$  code complete

```

int fact (int n) {
    int f = 1;
    (loop)
    return f;
}

```

```

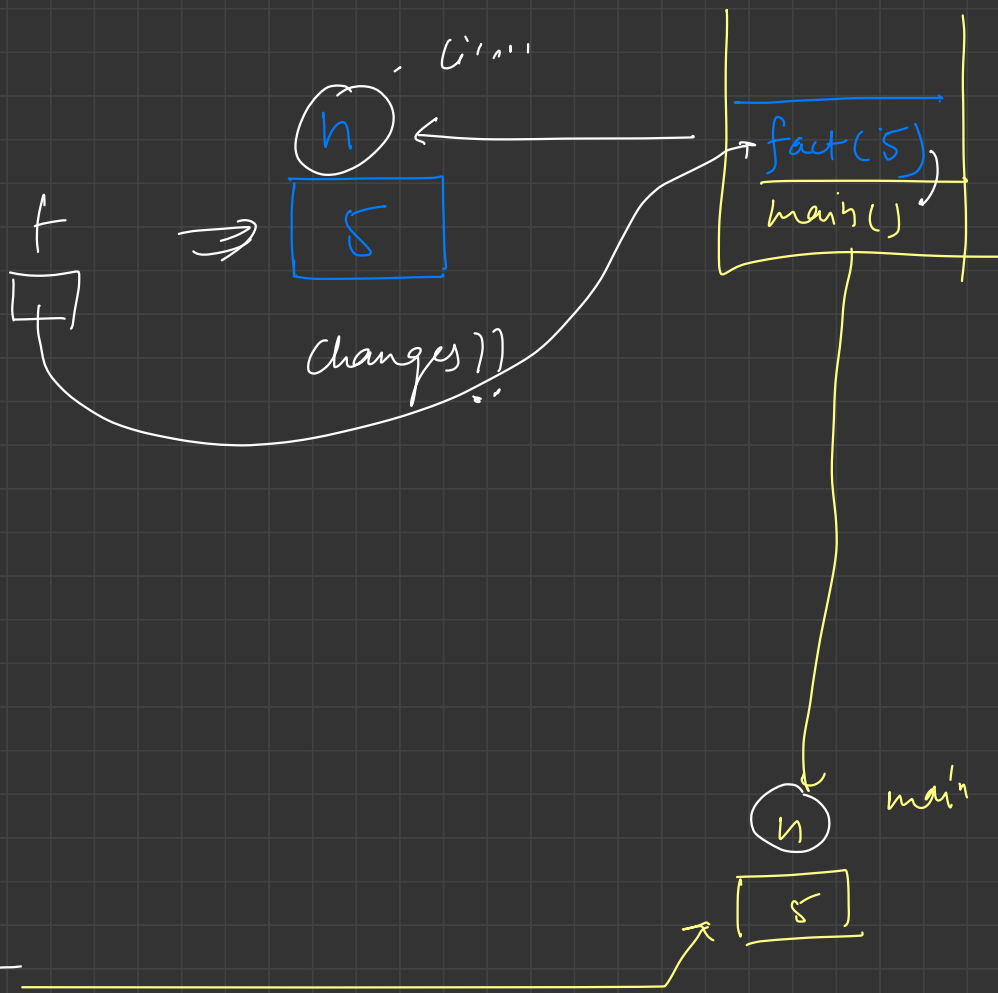
int main () {

```

```

    int n;
    cin >> n;
    cout << fact(n);
}

```



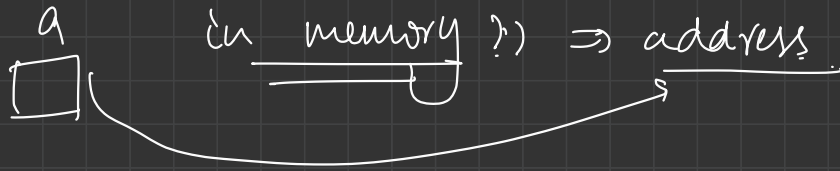
Functions are always called by value.

↓

- When a func. is called the values are passed to it.
- changes are not related to the variables of the caller.
- copies the values to its own variables

Q. Can we swap using a function ??

main      <sup>a</sup>  
□      <sup>b</sup>  
□      ⇐ refer



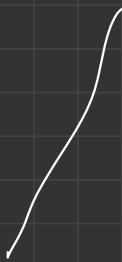
pass address to the function  
and ask the changes there ??

Will it be possible??

⇒ Call by reference.

- no copy
- original value available

Call by reference





Scope

Pointer  $\rightarrow$  function  $\rightarrow$  array  
param

function / local scope ?

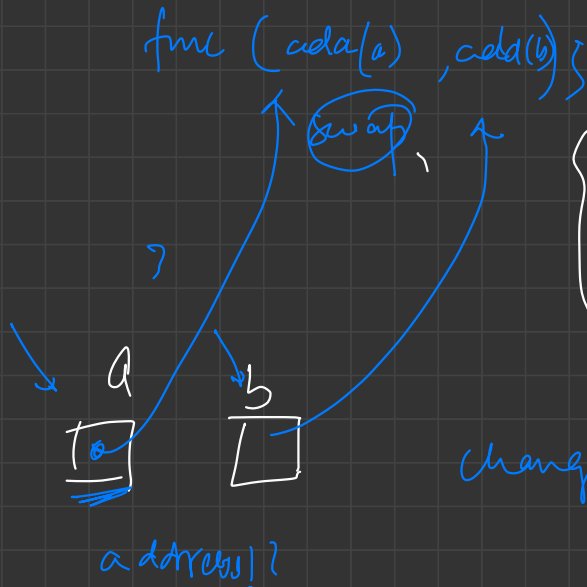
global scope ?

for ( i ; ; ) {  
    --  
    --  
    --  
} scope

{  
(  
if (     ) {  
    }  
}  
scope

scope  
↓  
↓

block scope {     }     }



{ Q. Print primes from 2 to N.

↓

a function to check prime ( ).

} changes main reflect?!

{ Pointers

call by reference

Passing arrays

}