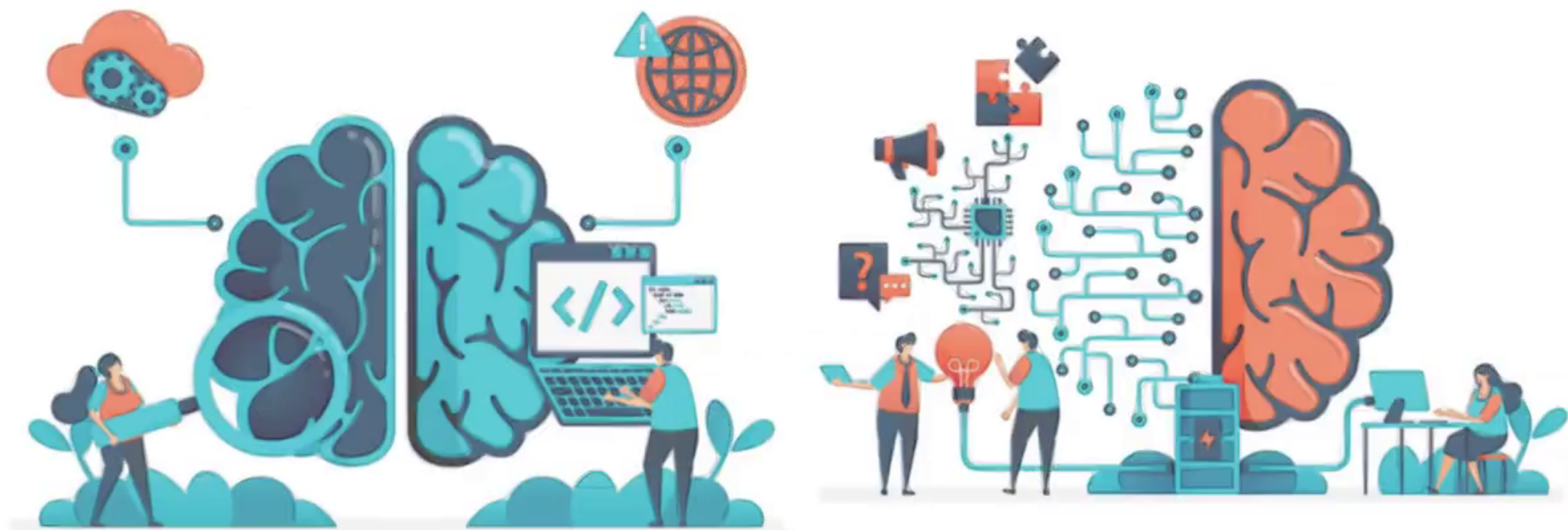


HOW DOES RECURSION REMEMBER STUFF ? (ARABIC INTUITION)



prior knowledge

1-my recursion playlist (on my channel)

2-stack data structure



Recursion tree

```
int main()  
{
```

```
    int a;  
    long long b;  
    char c;  
    double d;  
    string e;
```

```
    cout<<"i love anime";
```

```
    return 0;  
}
```



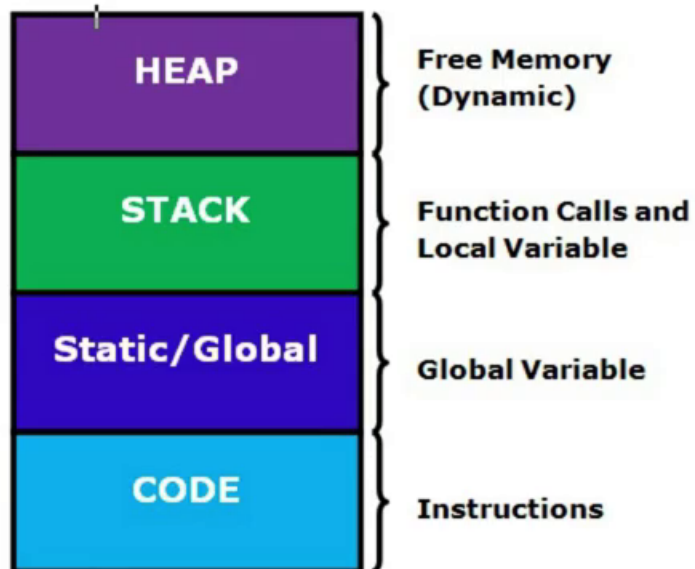
Data



Instructions

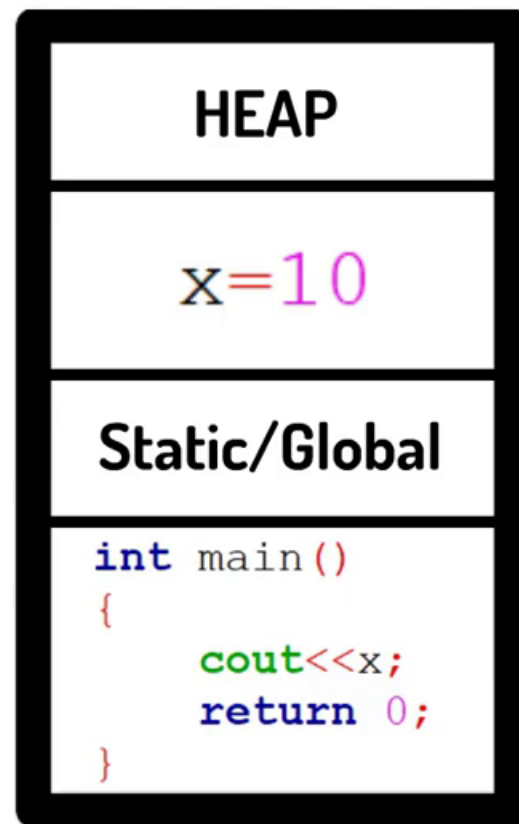
Recursion memory

Application Memory



www.binaryupdates.com

```
int main()  
{  
    int x=10;  
    cout<<x;  
    return 0;  
}
```



Recursion Tree Visualizer

Pre-defined templates

Fibonacci

Global variables

Recursive function

python

def fn(n):
 if (n == 0 or n == 1):
 return n

 return fn(n-1) + fn(n-2)

Options

Enable step-by-step animation

Enable memoization

Enable dark mode

fn(5)

Run

fn(5) starts running

5

memory stack

rectangle-frame.png

8/11/2021

Recursion Tree Visualizer

Pre-defined templates

Fibonacci

Global variables

Recursive function

python

def fn(n):
 if (n == 0 or n == 1):
 return n

 return fn(n-1) + fn(n-2)

Options

Enable step-by-step animation

Enable memoization

Enable dark mode

fn(5)

Run

fn(5) starts running

5

memory stack

main()

Made with ♥ by Bruno Papa • Github

rectangle-frame.png

Type here to search

92°F Clear 8:49 PM 8/11/2021

Recursion Tree Visualizer

Pre-defined templates

Fibonacci

Global variables

Recursive function

python

def fn(n):
 if (n == 0 or n == 1):
 return n

 return fn(n-1) + fn(n-2)

Options

Enable step-by-step animation

Enable memoization

Enable dark mode

fn(5)

Run

fn(4) starts running

5

4

memory stack

fib(5)

main()

rectangle-frame.png

Show all

Type here to search

92°F Clear

8:50 PM

8/11/2021

Recursion Tree Visualizer

Pre-defined templates

Fibonacci

Global variables

Recursive function

python

```
def fn(n):  
    if (n == 0 or n == 1):  
        return n  
  
    return fn(n-1) + fn(n-2)
```

Options

Enable step-by-step animation

Enable memoization

Enable dark mode

fn(5)

Run

fn(3) starts running

```
graph TD; 5((5)) --> 4((4)); 4 --> 3((3)); style 3 fill:#007bff,color:#fff
```

memory stack

```
graph TD; subgraph stack [memory stack]; direction TB; main[main()]; fib5[fib(5)]; fib4[fib(4)]; end
```

Made with ♥ by Bruno Papa • Github

rectangle-frame.png

Show all

Type here to search

92°F Clear

8:50 PM

8/11/2021

Recursion Tree Visualizer

Pre-defined templates

Fibonacci

Global variables

Recursive function

python

```
def fn(n):  
    if (n == 0 or n == 1):  
        return n  
  
    return fn(n-1) + fn(n-2)
```

Options

Enable step-by-step animation

Enable memoization

Enable dark mode

fn(5)

Run

fn(2) starts running

```
graph TD  
    5((5)) --> 4((4))  
    4 --> 3((3))  
    3 --> 2((2))  
    style 2 fill:#007bff,color:#fff
```

memory stack

```
graph TD  
    subgraph stack [memory stack]  
        fib3[fib(3)]  
        fib4[fib(4)]  
        fib5[fib(5)]  
        main[main()]  
    end
```

Made with ♥ by Bruno Papa • Github

rectangle-frame.png

Type here to search

92°F Clear 8:50 PM 8/11/2021

Recursion Tree Visualizer

Pre-defined templates

Fibonacci

Global variables

Recursive function

python

```
def fn(n):  
    if (n == 0 or n == 1):  
        return n  
    return fn(n-1) + fn(n-2)
```

Options

Enable step-by-step animation

Enable memoization

Enable dark mode

fn(5)

Run

fn(1) starts running

memory stack

Made with ♥ by Bruno Papa • Github

rectangle-frame.png

Show all

Type here to search

92°F Clear

8:50 PM

8/11/2021

Recursion Tree Visualizer

Pre-defined templates

Fibonacci

Global variables

Recursive function

python

def fn(n):
 if (n == 0 or n == 1):
 return n

 return fn(n-1) + fn(n-2)

Options

Enable step-by-step animation

Enable memoization

Enable dark mode

fn(5)

Run

fn(1) returns 1 to fn(2)

```
graph BT; 1((1)) --> 2((2)); 2 --> 3((3)); 3 --> 4((4)); 4 --> 5((5))
```

memory stack

fib(1)
fib(2)
fib(3)
fib(4)
fib(5)
main()

Made with ♥ by Bruno Papa • Github

rectangle-frame.png

Type here to search

92°F Clear 8:50 PM 8/11/2021

Recursion Tree Visualizer

Pre-defined templates

Fibonacci

Global variables

Recursive function

python

```
def fn(n):  
    if (n == 0 or n == 1):  
        return n  
  
    return fn(n-1) + fn(n-2)
```

Options

Enable step-by-step animation

Enable memoization

Enable dark mode

fn(5)

Run

fn(2) continues running

```
graph BT; 1((1)) -- 1 --> 2((2)); 2 --> 3((3)); 3 --> 4((4)); 4 --> 5((5)); style 2 fill:#007bff,stroke:#007bff,stroke-width:2px
```

memory stack

```
graph BT; main[main()] --> fib5[fib(5)]; fib5 --> fib4[fib(4)]; fib4 --> fib3[fib(3)]; fib3 --> fib2[fib(2)];
```

Made with ♥ by Bruno Papa • Github

rectangle-frame.png

Show all

Type here to search

92°F Clear 8:50 PM 8/11/2021

Recursion Tree Visualizer

Pre-defined templates

Fibonacci

Global variables

Recursive function

python

```
def fn(n):  
    if (n == 0 or n == 1):  
        return n  
    return fn(n-1) + fn(n-2)
```

Options

Enable step-by-step animation

Enable memoization

Enable dark mode

fn(5)

Run

fn(0) returns 0 to fn(2)

```
graph TD  
    5((5)) --> 4((4))  
    4 --> 3((3))  
    3 --> 2((2))  
    2 --> 1((1))  
    2 --> 0((0))  
    1 --> 2  
    0 --> 2
```

memory stack

```
graph TD  
    fib0[fib(0)]  
    fib2[fib(2)]  
    fib3[fib(3)]  
    fib4[fib(4)]  
    fib5[fib(5)]  
    main[main()]
```

Made with ♥ by Bruno Papa • Github

rectangle-frame.png

8/11/2021

Recursion Tree Visualizer

Pre-defined templates

Fibonacci

Global variables

Recursive function

python

```
def fn(n):  
    if (n == 0 or n == 1):  
        return n  
  
    return fn(n-1) + fn(n-2)
```

Options

Enable step-by-step animation

Enable memoization

Enable dark mode

fn(5)

Run

fn(2) continues running

```
graph TD  
    5((5)) --> 4((4))  
    4 --> 3((3))  
    3 --> 2((2))  
    2 --> 1((1))  
    2 --> 0((0))  
    style 2 fill:#007bff,color:#fff
```

memory stack

```
graph BT  
    main["main()"]  
    fib5["fib(5)"]  
    fib4["fib(4)"]  
    fib3["fib(3)"]  
    empty1[" "]  
    empty2[" "]  
    main --- fib5 --- fib4 --- fib3 --- empty1 --- empty2
```

Made with ♥ by Bruno Papa • Github

rectangle-frame.png

8:51 PM 8/11/2021

Pre-defined templates
Fibonacci

Global variables

Recursive function python

```
def fn(n):  
    if (n == 0 or n == 1):  
        return n  
    return fn(n-1) + fn(n-2)
```

Options
Enable step-by-step animation ☒
Enable memoization ☐
Enable dark mode ☐

fn(5) Run

