INTRODUCTION

STATIC VS DYNAMIC MEMORY ALLOCATION

```
void main()
{
        int  a;
        float b;
}
```

65535

Heap

STACK

Stack frame

Activation record

a	b
2	4

main ()

Code section

0
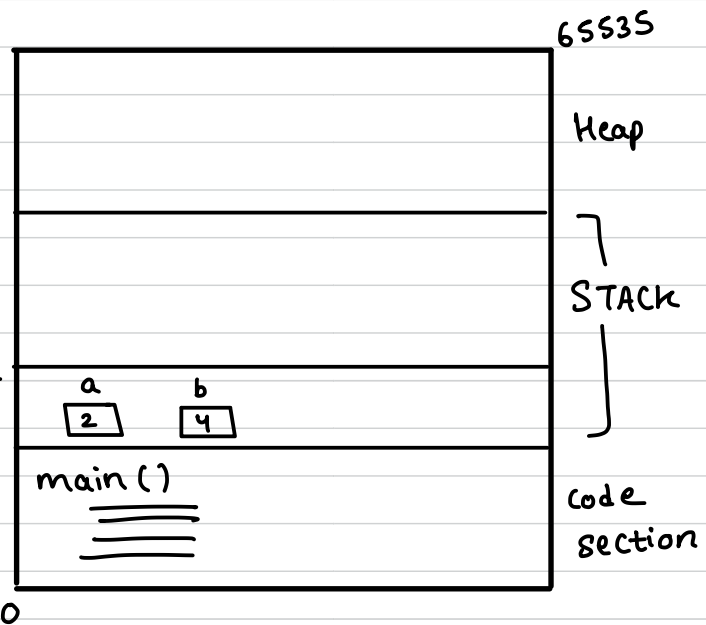
HOW DOES STACK WORK?

```
void fun2( int i)
{
        int  a;


}


void fun1 ()
{
    int x;
    fun2(x);
}


void main()
{
        int a;
        float b;
        fun1();
}
```

Heap

fun2 | i | a |

fun1 | x |

main | a | b |

fun2()   fun1()   main()

Stack

Code Section
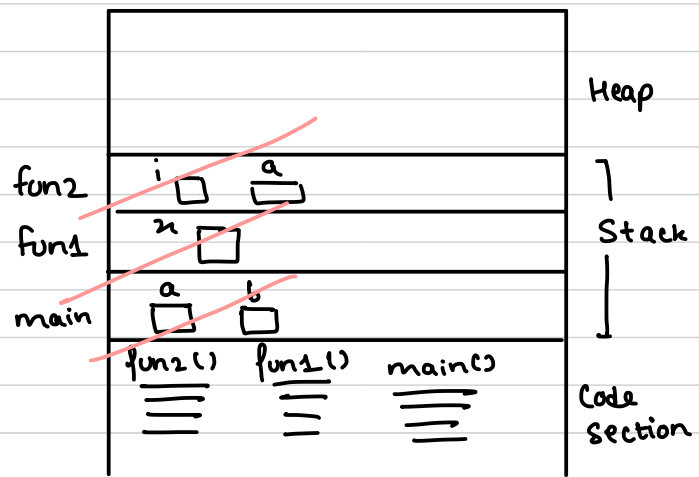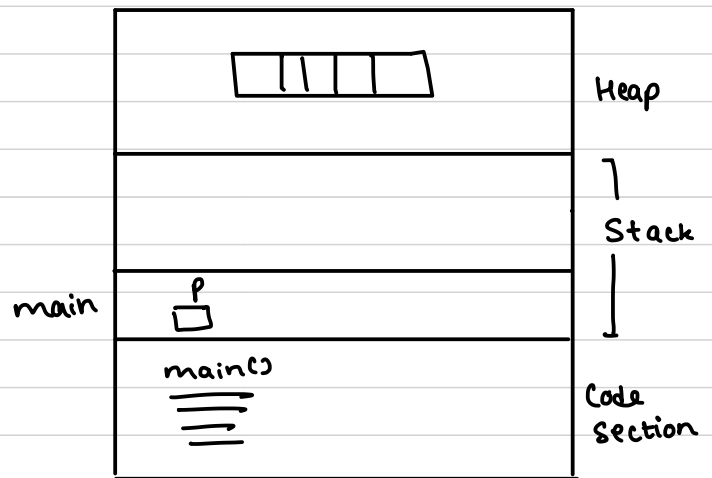
1. Stack is always organised memory.

HOW DOES HEAP MEMORY WORK?

1. Heap may be organised memory or unorganised memory.
2. Heap must be treated as a resource i·e when it is needed, we must use it and when not needed, free it so that it can be used by other applications.
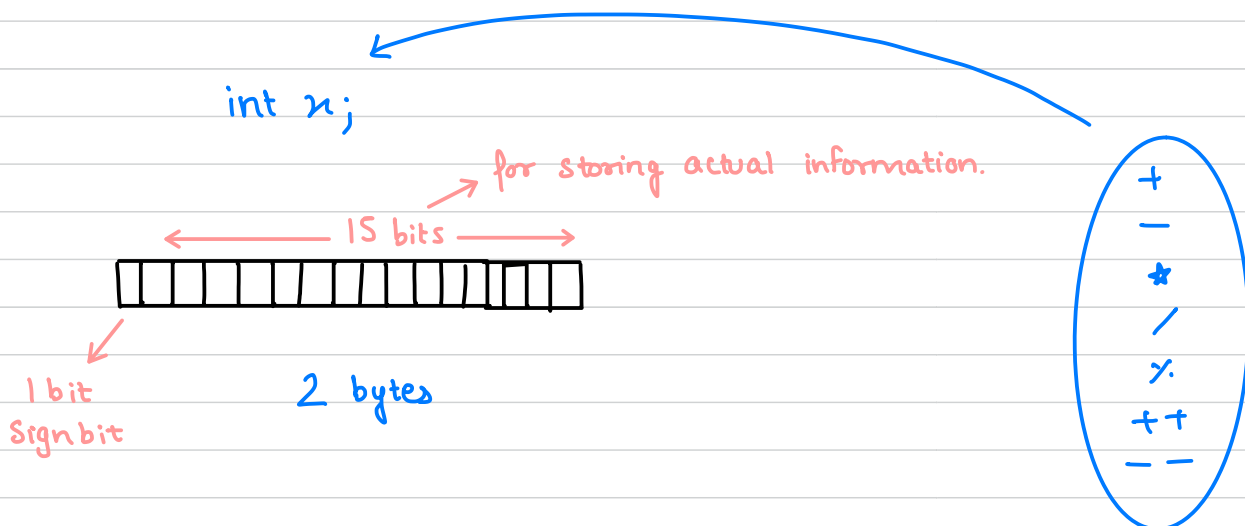   For eg — printer is a resource.

```
void main()
{
        int *p;

for C++   p= new int [5];
  for C   p= (int*) malloc (2*5);
                    └─→ for array
DEALLOCATION  delete []p;
        p = NULL;

}
```

main

	Heap
🔲	
	⌐ Stack
p	
▭	
main()	Code
≡	Section

DATATYPE?

   ↳   1. Representation of data.
        2. Operation on data.

int n;

                  ↗ for storing actual information.

←——— IS bits ———→

▭▭▭▭▭▭▭▭▭▭▭▭▭▭▭▭

↙
1 bit   2 bytes
Sign bit

(
+
−
*
/
%
++
−−
)

ABSTRACT DATATYPE

↳ hiding internal details

(part of object oriented programming)

What does log mean?

$\log_2 n \longrightarrow$ This gets divided by this until it reaches 1.

Time and Space complexity

```
void swap(x,y)
{
        int t;
        t = x;          ———  1
        x = y;          ———  1
        y = t;          ———  1
                             3
}
```

O(1)

```
int sum (int A[], int n)
{
        int s,i;
        s = 0;                    ———      1
        for (i=0; i<n; i++)       ———    n+1
        {          1        n
                s = s + A[i];     ———      n
        }
        return s;                 ———      1
}                                        2n+3
```

because i will be initialized (+1) and
i will be incremented for n no of times
and 1 time it will fail too (+n)

// because one time it will fail too.

O(n)

```
Void   Add (int n)
{
        int i, j;
        for (i=0 ; i<n; i++)                    ———————— n+1
        {
            for (j=0 ; j<n ; j++)   ———————   n*(n+1)    because the nested loop
            {                                             will run again and
                c[i][j] = A[i][j] + B[i][j] ;  ─  n*n    again for (n+1) no of
            }                                             times.
        }
}                           f(n) = 2n² + 2n + 1

                            O(n²)
```

The equations in the notes:

$$n+1$$

$$n * (n+1)$$

$$n * n$$

$$f(n) = 2n^2 + 2n + 1$$

$$O(n^2)$$