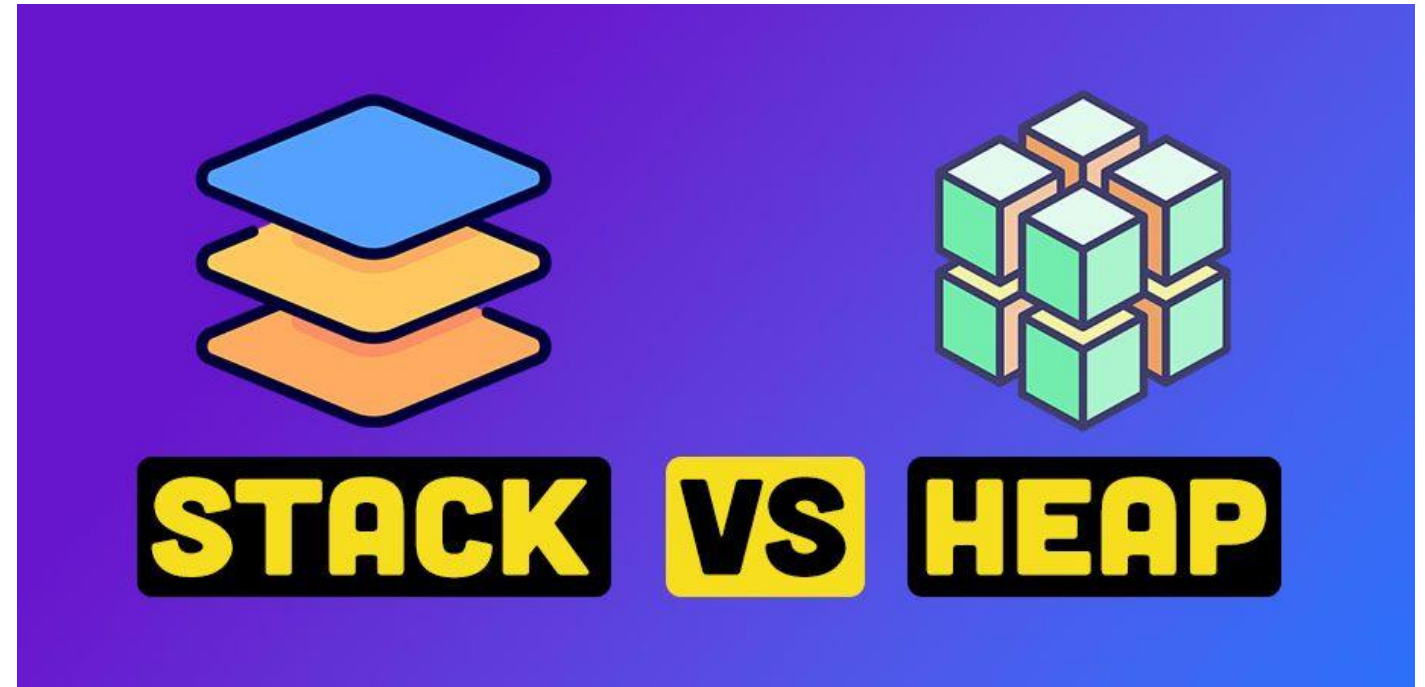# STACK & HEAP

Detailed analysis

What are they?

- **Stack:** A region of memory automatically managed by the compiler, used to store local variables and data related to function calls.

- **Heap:** A region of memory managed manually by the programmer (in C) or automatically by the garbage collector (in C#), used for dynamic storage of data.

Examples:

Stack:

```
1   #include <stdio.h>
2
3       int Add(int a){
4           int x = 10; //local variable allocated in the STACK
5           return a+x;
6       }
7
8   int main(){
9
10      int a = 4;
11
12      a = Add(a); //function call for information contained in the STACK
13      printf("%d\n", a);
14
15
16      return 0;
17  }
```
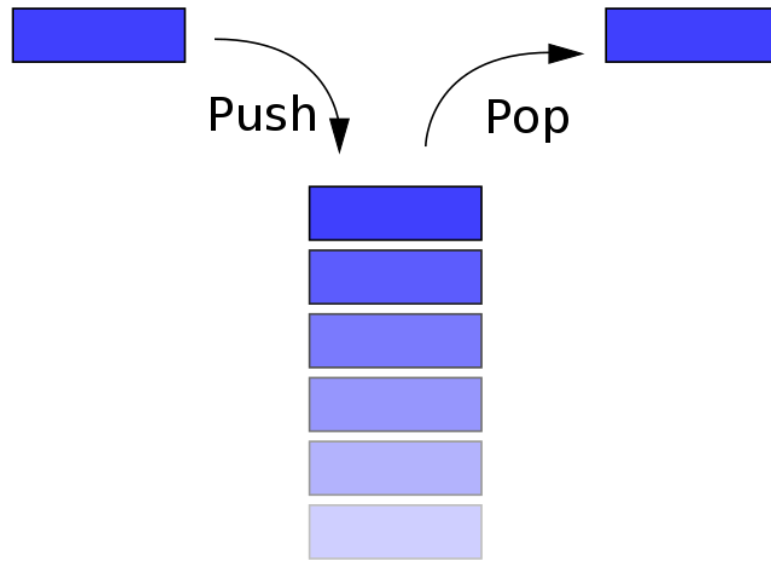
Heap:

```
int *p = (int*)malloc(sizeof(int)); //manual memory allocation
free(p); //manual memory deallocation
```

What they are useful for?

Stack: automatically manages the life of local variables and provides a function call handling mechanism.

Heap: Unlike the stack, which is limited in size and automatically manages local variables, the heap allows memory to be allocated more flexibly.

Management policies:

Stack: Management is automatic, with the variables being allocated and deallocated in LIFO (Last In, First Out) mode, the last variable to be allocated is the first to be deallocated.

Heap: In C, management is manual, the programmer must allocate and deallocate memory. In C# however, management is automatic and takes place via the garbage collector.

Stack (C and C#): Local variables, return addresses, function parameters.

Heap (C): Data dynamically allocated with malloc() or new.

Heap (C#): Objects and data allocated with new, automatically managed by the garbage collector.

What is stored?

Example diagram

This example represents the **STACK**, which operates using LIFO (Last In First Out) and which is necessary for the compiler's allocation of static variables, and the **HEAP** which instead sees the manual allocation in C and automatic allocation in C# with the garbage collector of static variables.

Ordered, on top of eachother!

No particular order!

Stack

Heap