**Mohammad Seyfi Marandi - Exercise 3**

**Stack Memory**

* **Definition**: Stack memory is a region of memory that is used for static memory allocation, primarily for storing function call information, local variables, and other programmatic data during the execution of a program.
* **Characteristics**:
  + Follows a last-in, first-out (LIFO) structure, where the last item pushed onto the stack is the first to be popped off.
  + Managed automatically by the compiler or runtime environment.
  + Memory allocation and deallocation are fast.
  + Typically has a fixed size determined at compile time.
  + Used for storing local variables, function parameters, return addresses, and execution context during function calls.
* **Lifetime**: Short-lived. Variables allocated on the stack are automatically deallocated when they go out of scope or when the function call ends.
* **Usage**: Well-suited for managing function calls and local variables with predictable lifetimes.

**Heap Memory**

* **Definition**: Heap memory is a region of memory that is used for dynamic memory allocation, where memory allocation and deallocation can occur in a random order. It is typically used for storing objects and data structures whose size or lifetime cannot be determined at compile time.
* **Characteristics**:
  + Follows a more complex memory management model compared to the stack.
  + Managed manually by the programmer or through garbage collection mechanisms.
  + Memory allocation and deallocation can be slower compared to the stack.
  + Typically has a larger size compared to the stack.
  + Can suffer from memory fragmentation due to arbitrary allocation and deallocation patterns.
* **Lifetime**: Can be long-lived. Memory allocated on the heap remains allocated until explicitly deallocated by the programmer or until the program terminates.
* **Usage**: Used for managing dynamic data structures like arrays, linked lists, objects created with **new** or **malloc**, and any data that needs to persist beyond the scope of a single function call.

**Summary**

* **Stack**: Static memory allocation, follows a LIFO structure, managed automatically, short-lived, used for local variables and function call information.
* **Heap**: Dynamic memory allocation, memory allocation and deallocation in any order, managed manually or through garbage collection, long-lived, used for dynamic data structures

**Stack Memory**

* **Structure**: Follows a last-in, first-out (LIFO) structure, meaning the last item pushed onto the stack is the first to be popped off.
* **Management**: Managed automatically by the compiler or runtime environment.
* **Speed**: Memory allocation and deallocation are fast.
* **Size**: Typically has a fixed size determined at compile time.
* **Lifetime**: Short-lived. Variables allocated on the stack are automatically deallocated when they go out of scope or when the function call ends.
* **Usage**: Well-suited for managing function calls and local variables with predictable lifetimes.

**Heap Memory**

* **Structure**: Memory allocation and deallocation can occur in a random order.
* **Management**: Managed manually by the programmer or through garbage collection mechanisms.
* **Speed**: Memory allocation and deallocation can be slower compared to the stack.
* **Size**: Typically has a larger size compared to the stack.
* **Fragmentation**: Can suffer from memory fragmentation due to arbitrary allocation and deallocation patterns.
* **Lifetime**: Can be long-lived. Memory allocated on the heap remains allocated until explicitly deallocated by the programmer or until the program terminates.
* **Usage**: Used for managing dynamic data structures like arrays, linked lists, objects created with **new** or **malloc**, and any data that needs to persist beyond the scope of a single function call.