EXPLAIN ABOUT STORAGE CLASSES

Storage classes in C define the scope, lifetime, visibility, and initial value of variables and functions. They tell the compiler where the variable is stored, how long it exists, and who can access it.

C provides four main storage classes:

1. auto
2. register
3. static
4. extern

1. Auto Storage Class

Auto is the default storage class for all local variables declared inside a function or block. These variables are created when the function is entered and destroyed when the function exits.

Characteristics:

- Scope: Local to the block
- Lifetime: Until the block ends
- Default value: Garbage value
- Stored in: Memory (RAM)

Example:

```
void func() {
   auto int x = 10;   // auto keyword is optional
}
```

2. Register Storage Class

The register storage class requests the compiler to store the variable in a CPU register instead of memory for faster access. It is mainly used for frequently accessed variables like loop counters.

Characteristics:

Scope: Local to the block

Lifetime: Until the block ends

Default value: Garbage value

Stored in: CPU register (if available)

- Address of a register variable cannot be accessed using &.
- Compiler may ignore the request if registers are not available.

Example:

```
void func() {
   register int i;
}
```

3. Static Storage Class

The static storage class preserves the value of a variable between function calls. Unlike local variables, static variables are initialized only once and exist for the entire program execution.

(a) Static Local Variable

Characteristics:

- Scope: Local to the function
- Lifetime: Entire program
- Default value: Zero

Example:

```
void func() {
   static int count = 0;
   count++;
   printf("%d\n", count);
}
```

(b) Static Global Variable

When a global variable is declared as static, it is accessible only within the same file, providing data hiding.

Example:

```
static int x = 10;
```

4. Extern Storage Class

The extern storage class is used to declare a global variable or function defined in another file. It allows sharing variables across multiple source files.

Characteristics:

- Scope: Global
- Lifetime: Entire program
- Default value: Zero

Example:

```
// file1.c
int x = 10;

// file2.c
extern int x;
```