Storage classes

Wednesday, 10. August 2022 04:29pm

The storage class in c++ is used to access object of a program in various scope

static

The static storage class is use to create an object (example == **functions variables enums** ==) that will only exist in the same module (source file) or in the function that it was created in

```
int main(){
  int number;
  return 0;
}

int test(){
  static int number;
  number = 3;
  return number;
}
```

In the above code, the variable **number** is created and accessed in **function scope** and is not affect by any variable object with the same name outside that function.

properties

- They do not have initial values (stays undifined when initialized)
- Advisable to initialize it on declaration.

  ```
  static int number = 2;
  ```

- Access is restricted to only the **file** in which it was declared or the **function** in which it was declared.

---

extern

The extern storage class is used to access any object that is declared outside any function (**Global scope**)

Global scope:

any object or that is initialized outside any funciton. A global scope object is only initialized without defining it. And therefore modified through out the program Example

```
//file test_1.cpp of test::program
#include <iostream>
#include <string>

std::string line;   //global variable

int main(){
  line = "this is a line";
  return 0;
}
```

.

```
//file test_2.cpp of test::program
#include <iostream>
#include <string>

extern string line; // this will access the variable from the test_1.cpp module
```

In the above code, the the **line** variable is now access in the second file of the **SAME PROGRAM**

The **extern** storage class is used explicitly to import the a global object (example: **function**, **variable**, **enums**) from another module or file of the same program.

Properties

- They are used to access a global object **In the same program**
- It is adviced not to initialize the object when declared by the extern storage class.

---

auto

The auto storage class is that is all of those objects defined in a function apart from that of a **static** storage class.

In relation to the **static** storage class the **auto** storage class will be destroyed on completion of the function.

```
void func (){
  int num = 2;
  }
```

properties

- All variables or object in a function are termed to stored in the auto storage class type.
- The **auto** keyword can be ommitted when creating an auto keyword class.
- No default value is assign when initialized.
- It is adviceable to always defined the **auto** storage type object.

---

register

The **register** storage keyword can be used to save a lot of memory in a c++ program. This storage class is used to store variables or objects that are of almost the same size of an int. Hence they are used to store very small type of variables.

Properties.

- The keyword **register** must preceed the datatype of the varible or object.

  ```
  register int num = 3;
  ```

- When no datatype is passed to the variable, the default used is an int. register num = 3; // default to int.

- Advicable to use only 1 int
  i. char
  ii. and pointers.