Introduction to structures and typedef

hitbullseye

Introduction to typedef

- typedef is a keyword that gives a new name to an existing data type. For example, you can give a new name to an existing data type unsigned int type and program using uint in its place. At this time, uint will also be treated exactly as unsigned int
- Basic Syntax :

  typedef <existing_data_type> <new_data_type>

Typedef of Struct

- When writing a code we certainly at most times have felt the need to define our own data type as per the requirement of the problem. In C language this can be achieved using two keywords: struct and typedef. These keywords help us to group non-homogeneous data types into a single group.
- The question remains why we need typedef if we still have struct?
  - Putting it simply, typedef means we no longer have to write struct every time in our code. Hence it makes our code look clean and readable. It also makes the code portable and easily manageable.

Typedef and its uses

- the typedef keyword helps user to provide meaningful names to the already existing data types in the C language and make it more understandable for others as well.
- For example , if you want to name int newly as MyInt , write as follows.

```
typedef int MyInt;
```

- So, now instead of using int we can also use MyInt as a substitute name for this data type.

Compilation Errors

- A compilation error may arise in the following cases :
  - ; missing at the end
  - 1st and 2nd designations are reversed
  - first type name does not exist
  - The second type name specifies a type name that already exists
  - Invalid characters are used in the second type name
  - You can't use characters that probably aren't allowed in variable names
  - no space

Advantages of Typedef

- Easy to read source code - With typedef You can make the source code easier to read by creating a type that is easy to understand.
- Easier to write types - typedef by shortening the new type name, it becomes easier to write that type.
- Easier to modify – It becomes easier to modify the new code.

Typedef Examples

- Examples of typedef definitions
- The following statements define LENGTH as a synonym for int and then use this typedef to declare length, width, and height as integer variables:

```
typedef int LENGTH; LENGTH length, width, height;
```

- The following declarations are equivalent to the above declaration:

```
int length, width, height;
```

Typedef Examples

- Similarly, typedef can be used to define a structure, union, or C++ class. For example:

```
typedef struct {
    int scruples;
    int drams;
    int grains;
    } WEIGHT;
```

Structures

- Structures (also called structs) are a way to group several related variables into one place. Each variable in the structure is known as a **member** of the structure.
- Unlike an array a structure can contain many different data types (int, float, char, etc.).

Creating a Structure

- You can create a structure by using the struct keyword and declare each of its members inside curly braces:

```
struct MyStructure {   // Structure declaration
    int myNum;          // Member (int variable)
    char myLetter;      // Member (char variable)
}; // End the structure with a semicolon
```

- To access the structure, you must create a variable of it.
- Use the struct keyword inside the main() method, followed by the name of the structure and then the name of the structure variable:

Why Structs in C

- Suppose you want to store information about a person: his/her name, citizenship number, and salary. You can create different variables name, citNo and salary to store this information. What if you need to store information of more than one person? Now, you need to create different variables for each information per person: name1, citNo1, salary1, name2, citNo2, salary2, etc. A better approach would be to have a collection of all related information under a single name Person structure and use it for every person.

Structs vs Typedef

- Structs define a new user-defined data type in C by grouping together multiple variables of different data types within a single structure.
- Typedef creates a new name for an existing data type in C, allowing developers to define more concise and intuitive names for complex data types.
- Structs provide a way to encapsulate related data and can be used to represent entities with multiple attributes, such as a person with name, age, and address.

Structs vs Typedef

- Typedef allows for the creation of custom aliases for data types, improving code readability and reducing complexity when working with complex or lengthy type names.
- Structs facilitate the creation of composite data structures, while typedef simplifies the process of declaring variables of existing data types with more descriptive names.

THANK YOU

hitbullseye