Variable : Memory Concept

Programs shown in the previous section print text on the screen. This section will introduce the concept of variable so that our program can perform calculation on data.

Program : Adding two numbers

We'll solve this problem in C++ with the following steps:

**STEP 1**: Allocate memory for storing three numbers
**STEP 2**: Store first number in computer memory
**STEP 3**: Store second number in computer memory
**STEP 4**: Add these two numbers together and store the result of the addition in a third memory location
**STEP 5**: Print the result

**STEP 1**: Now first we'll allocate memory for storing numbers.

Location of the computer memory which is used to store data and is given a symbolic name for reference is known as variable. We need three variables, two for storing input and third for storing result. Before a variable is used in a program, we must declare it. This activity enables the compiler to make available the appropriate type of location in the memory.

Following statements declare three variables of type integer to store whole numbers.

```cpp
int x;
int y;
int z;
```

You can declare more than one variable of same type in a single statement like:

```cpp
int x, y, z;
```

**STEP 2**: Following statement stores value in first variable
x = 25;

**STEP 3**: Following statement stores value in second variable

y = 10;

**STEP 4**: Now, add these two numbers together and store the result of the addition in third variable

z = x + y;

**STEP 5**: Print the result

cout << "The sum is ";
cout << sum;

You can combine above two statements in one statement

cout << "The sum is " << sum;

here, is the complete program

```cpp
#include <iostream>
using namespace std;

int main()
{
    //declare variables of integer type
    int x;
    int y;
    int z;

    //storing value in variables
    x = 25;
    y = 10;

    //adding numbers and store the result in sum
    z = x + y;

    //print the result
    cout << "The sum is ";
cout << z;

    return 0;
}
```

Output: The sum is 35
Identifiers

Symbolic names can be used in C++ for various data items used by a programmer in his program. A symbolic name is generally known as an identifier. The identifier is a sequence of characters taken from C++ character set. In previous program x, y and z are identifiers of variables. The rule for the formation of an identifier are:

- An identifier can consist of alphabets, digits and/or underscores.
- It must not start with a digit
- C++ is case sensitive that is upper case and lower case letters are considered different from each other.
- It should not be a reserved word.

Keywords

There are some reserved words in C++ which have predefined meaning to compiler called keywords. These words may not be used as identifiers. Some commonly used Keywords are given below:

<table>
<thead>
<tr>
<th>asm</th>
<th>auto</th>
<th>bool</th>
<th>break</th>
<th>case</th>
</tr>
</thead>
<tbody>
<tr>
<td>catch</td>
<td>char</td>
<td>class</td>
<td>const</td>
<td>const_cast</td>
</tr>
<tr>
<td>continue</td>
<td>default</td>
<td>delete</td>
<td>do</td>
<td>double</td>
</tr>
<tr>
<td>dynamic_cast</td>
<td>else</td>
<td>enum</td>
<td>explicit</td>
<td>export</td>
</tr>
<tr>
<td>extern</td>
<td>false</td>
<td>float</td>
<td>for</td>
<td>friend</td>
</tr>
<tr>
<td>goto</td>
<td>if</td>
<td>inline</td>
<td>int</td>
<td>long</td>
</tr>
<tr>
<td>mutable</td>
<td>namespace</td>
<td>new</td>
<td>operator</td>
<td>private</td>
</tr>
<tr>
<td>protected</td>
<td>public</td>
<td>register</td>
<td>reinterpret_cast</td>
<td>return</td>
</tr>
<tr>
<td>short</td>
<td>signed</td>
<td>sizeof</td>
<td>static</td>
<td>static_cast</td>
</tr>
<tr>
<td>struct</td>
<td>switch</td>
<td>template</td>
<td>this</td>
<td>throw</td>
</tr>
<tr>
<td>true</td>
<td>try</td>
<td>typedef</td>
<td>typeid</td>
<td>typename</td>
</tr>
<tr>
<td>union</td>
<td>unsigned</td>
<td>using</td>
<td>virtual</td>
<td>void</td>
</tr>
<tr>
<td>volatile</td>
<td>wchar_t</td>
<td>while</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>