Tutorial Link https://codequotient.com/tutorials/Variables and Identifiers/5a1d944752795c1b16c0abe5

TUTORIAL

Variables and Identifiers

Chapter

1. Variables and Identifiers

Topics

1.2 Variable

1.3 Constants

1.7 Identifiers

In C/C++ programs, the most fundamental element is expression. Expression can be formed by mixing data and operators. Data can be represented by variables, constants etc. whereas operators are defined in every programming languages. C/C++ languages supports various data types. All these topics are explained below: -

Variable

A variable is a named location in memory. It is used to hold a value during the execution of program. All variables needs to be declared before they used. Every variable has a specific data type associated with it. Which shows the type of data it can hold. To declare a variable, we can use as below:

```
data_type var_name;
```

data_type is a valid data type and modifier and var_name is the name of the variable declared.

Constants

Constants refer to fixed values that the program may not alter. Constants can be of any of the basic data types. The way each constant is represented depends upon its type. Constants are also called literals.

Character constants are enclosed between single quotes. For example, 'a' and '%' are both character constants. Integer constants are specified as numbers without fractional components. For example, 10 and –100 are integer constants. Floating-point constants require the decimal point followed by the number's fractional component. For example, 11.123 is a floating-point constant. C also allows you to use scientific notation for floating-point numbers. For example,

C

```c
#include <stdio.h>

int main()
{
  char c = 'A';        // character constant
  int i = 50;        // integer constant
  double    df = 3.5;    // double constant

  printf("c = %c\n", c);
  printf("i = %d\n", i);
  printf("df = %lf\n", df);

  return 0;
}

```

C++

```cpp

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

```

int main()
{
  char c = 'A';        // character constant
  int i = 50;         // integer constant
  double    df = 3.5;    // double constant

  cout<<"c = "<<c<<endl;
  cout<<"i = "<<i<<endl;
  cout<<"df = "<<df;
  return 0;
}

```

When a constant value starts with 0, it is considered as octal number. So if we use as,

```
int a = 070;
```

and then want to print the value of a in decimal it is not 70, it is the decimal equivalent of 70 base 8 which is 56. Also following is a syntax error,

```
int a = 078;
```

as you are writing an octal constant, but in octal only digits from 0 to 7 can be used, hence 8 is not a valid digit in octal number system.

C

```
#include <stdio.h>
int main()
{
  int a = 070;
  printf("a = %d", a);

  return 0;
}
```

C++

```

#include<iostream>
```

```
#include<cstdio>
using namespace std;

int main()
{
  int a = 070;
  cout<<"a = "<<a;

  return 0;
}

```

Identifiers

In C/C++, the names of variables, functions, labels, and various other user-defined items are called identifiers. The length of these identifiers can vary from one to several characters. The first character must be a letter or an underscore, and subsequent characters must be either letters, digits, or underscores. Here are some correct and incorrect identifier names:

Correct : employee_name, emp_phone, _empid, _number123

Incorrect : 1empno, emp-number, first&last, high..bal

In an identifier, upper and lowercase are treated as distinct. Hence count, Count and COUNT are three separate identifiers.