Programming in C

Chapter -3

Variables and Data types

by, Santa Basnet

Variables and Data types

Before writing C program

The only way to learn a new programming language is by writing programs in it.

Variables and Data types

Before writing C program

Common Programming Errors:

Syntax Error : wrong sentences.

```
float x = ;
int marks sum;
```

Logical Error : wrong formula.

```
mean = a + b / 2;
```

Runtime Error : something not defined/not available.

```
divided by 0, unable to open file,
out of memory etc.
```

Variables and Data types

Your first C program

Hello World Program

```
/*This is my first program.*/
#include <stdio.h>
int main()
{
    printf("Hello, world!\n");
    return(1);
}
```

Your comments.

Standard library, preprocessor directive.

Program starts from here, a main function.

Start main

Your Text for output

Main function wants you to return an integer number.

End of main function.

Variables and Data types

Your first C program

Build and Run: Hello World Program

```
# cd your_working_directory
# gcc -o hello hello.c
# hello
  Hello, world!
```

C:\windows\system32\cmd.exe

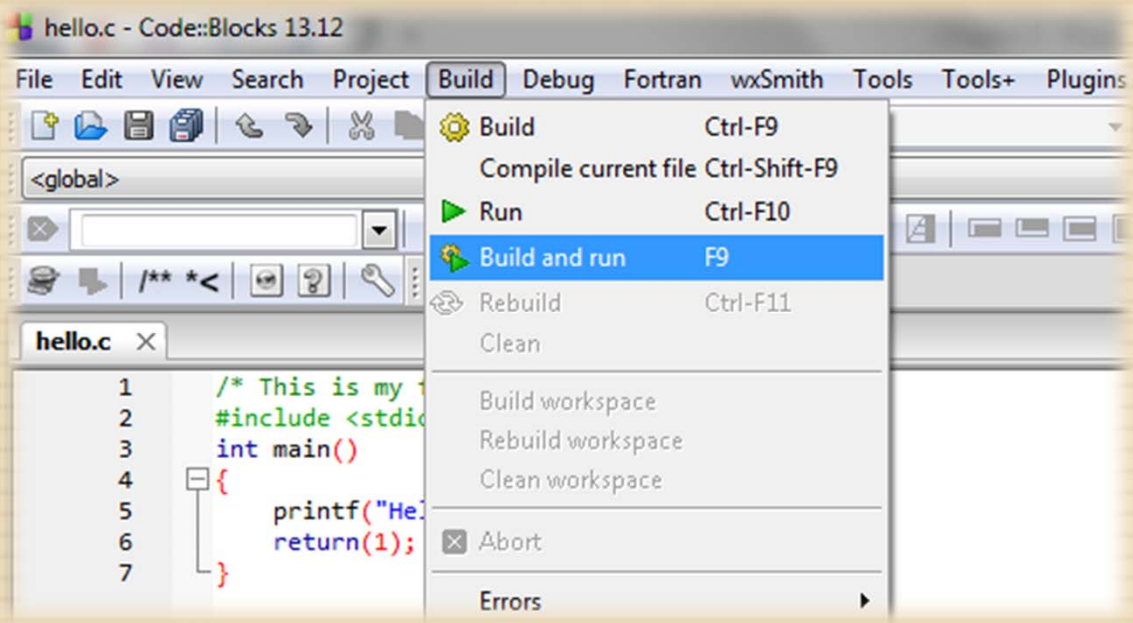
```
D:\>cd D:\nec Lectures\C\Lecture\Chapter 2\Examples

D:\nec Lectures\C\Lecture\Chapter 2\Examples>gcc -o hello hello.c

D:\nec Lectures\C\Lecture\Chapter 2\Examples>hello
Hello, world!

D:\nec Lectures\C\Lecture\Chapter 2\Examples>
```

Variables and Data types

Your first C program

Build and Run: Hello World Program

```
# cd your_working_directory
# gcc -o hello hello.c
# hello
  Hello, world!
```



Variables and Data types

Variables

A C program, whatever its size, consists of functions and variables.

Variables and Data types

Variable declaration

- All the variables must be declared before they are used and at beginning of the function.
- A variable declaration proclaims the properties of variables.
- A declaration consists of a type name and a list of variables.

Syntax: `type variable_list;`

Example:

```
int lower, upper, step;
float weight, height;
```

Variables and Data types

Data types

- C language allows you to declare 6 basic data types: (rely on the machine architecture)

1. int – variables are of integer type, 4 byte of data.
2. float – variables are of floating point, may have fractional part of at least 6 digits precision, 4 byte of data.
3. char – a single byte integer data, character.
4. long – variables are of integer, 4 byte of data.
5. double – variables are of floating point with higher precision with 15 decimal places. 8 byte of data.

Variables and Data types

Data types

- More data types that C supports:

6. short – variables are of integer type, 2 byte of data.

7. signed and unsigned – integer variables can be either signed or unsigned. singed type allows you to store negative integers too where as unsigned types stores only positive integers.

8. Even more types:

```
long long int x;/* 8 bytes */
long double y;  /* 12 bytes */
```

Variables and Data types

Data types

- More data types that C supports:

  7. More on signed and unsigned – An example

     # int type (4 byte) can store -2,147,483,648 to 2,147,483,647.

     # unsigned int type (4 byte) can store 0 to 4,294,967,295.

     # How these numbers appear here ?

Variables and Data types

Symbolic Constants

- C language allows you to give meaningful name for the numbers (constants) used in the program.
- A #define does this for you. It replaces all the occurrences of name by its replacement list during compilation.

Syntax: `#define name replacement`

Example:
```
#define LOWER 0
#define UPPER 32767
#define HELLO "Hello world!"
```

Variables and Data types

Symbolic Constants

- You can use const keyword in a variable name to declare constants too. Once you make a variable const, you cannot change the value of the variable i.e. a read only variable.
- Example:

```
const int UPPER = 32767;
const int LOWER = 0;
```

Variables and Data types

Symbolic Constants

- Some Constants.
- Integer type:

  `0, -9, 879 → decimal`

  `076, -077 → octal`

  `0x67A, 0x82F → hexa-decimal`
- Float type:

  `-2.0f, 0.0000234, -0.22E-5`
- Character type:

  `'A', 'u', '8'`
- String constant:

  `"Nepal", "    ", "A", ""`

Variables and Data types

Simple Input/Output

- Character input and output: we can read/write a character at a time by numerous functions through standard library.

  # with format specifier: %c

```
scanf("%c", &ch);
printf("%c", ch);
```

  # with getchar & putchar:

```
ch = getchar();
putchar(ch);
```

Variables and Data types

Simple Input/Output

- Integer and Floating point numbers:

  # int and long type with format specifier:

  ```
  %d or %i, %o, %x, %u & %ld.
  scanf("%d", &age);
  ```

  # floating point type with format specifier:

  ```
  %f, %g, %Lf, %e
  printf("%e", g);
  printf("%f", virus_size);
  ```

Variables and Data types

Simple Input/Output

- String type: various library functions and format specifier allows you to read/write string i.e. a sequence of characters.

  # with format specifier: %s

```
scanf("%s", &name);
printf("%s", name);
```

  # with gets & puts:

```
gets(name);
puts(name);
```

Variables and Data types

Simple Input/Output

- Escape sequences: represents special character within string and characters.

Escape Sequence	Description	Representation
\'	single quote	byte 0x27 in ASCII encoding
\"	double quote	byte 0x22 in ASCII encoding
\?	question mark	byte 0x3f in ASCII encoding
\\	backslash	byte 0x5c in ASCII encoding
\b	backspace	byte 0x08 in ASCII encoding
\f	form feed - new page	byte 0x0c in ASCII encoding
\n	line feed - new line	byte 0x0a in ASCII encoding
\r	carriage return	byte 0x0d in ASCII encoding
\t	horizontal tab	byte 0x09 in ASCII encoding

Variables and Data types

Simple Input/Output

- Putting all together:

```
/* Variable and Constant demonstration program. */
#include <stdio.h>
#define MSG "Enter Your Name : "
int main()
{
    char Name[100];
    const float PI = 3.14159f;
    const float G = 6.67e-11;
    int m_Char = 0x6D;
    printf(MSG);
    gets(Name);
    printf("Hello %s!\n",Name);
    printf("Character m -> %d\t%X\n", m_Char, m_Char);
    printf("Pi & G -> %2.4f\t%e\n", PI, G);
    return(1);
}
```

"D:\nec Lectures\C\Lecture\Chapter 2\Examples\dataconst.exe"

```
Enter Your Name : Rocky
Hello Rocky!
Character m -> 109      6D
Pi & G -> 3.1416        6.670000e-011
```

Variables and Data types

Operators

- Operators are used within an expression which specifies the manipulation of operands at the time of evaluation.
- C supports very rich set of operators.
- Operators have precedence and associativity.

  # unary : takes single operand

  # binary : takes two operands

  # ternary : takes three operands

Variables and Data types

Operators

- Arithmetic, Assignment and Augmented assignment:

```
# +, -, *, /, %
        a = b / c; x = y % z;
# =
        x = y;

# +=, -=, *=, /=, %=, &=, |=, ^=, <<=, >>=
        a += b; x &= z;
```

Variables and Data types

Operators

- Bitwise logic, Boolean logic and Bitwise shift:

```
# ~, &, |, ^
        a = b & c; x = ~y;
# !, &&, ||
        x = y && z; a = !b;
# <<, >>
        a = b << 2; x = z >> 3;
```

Variables and Data types

Operators

- Relational, equality and increment/decrement

```
# >, <, >=, <=
        a >= b && c < x
# !=, ==
        a = (b != c);
# ++, --
        a++; --x;
```

- Conditional Evaluation: ? :

```
a = b < 0 ? –b : b;
```

Variables and Data types

Operators

- Size, Calling function, Member selection.

```
# sizeof
        sizeof(b);
# ()
        readInput();
# ., ->
        c = a->b; y = x.c;
```

- Type conversion: (type_name)

```
int a = (int)b;
```

Variables and Data types

Operators

- Comma Operator: sequence of operations can be separated by comma.

```
# int a=1, b=2, c=3, i=0;
# i = (a += 2, a + b); //5
# i = (a, b, c); //3
```

Variables and Data types

Execution sequence

- Understanding the sequence of execution:

```
/*Swapping values of the variables*/
#include <stdio.h>
int main(){
    int x = 99, y = 11;
    x = x + y; /* x = 110, y = 11 */
    y = x - y; /* x = 110, y = 99 */
    x = x - y; /* x = 11, y = 99 */
    printf("x = %d, y = %d", x, y);
    return(1);
} /* Final Output: x = 11, y = 99 */
```

Thank you.

Questions ?