URBAN EDGE

Chapter 1: Variables, Constants & Keywords

Variables

A variable is a container which stores a 'value'. In Kitchen, we have containers storing Rice, Dal, Sugar etc. Similar to that variables in C stores value of a constant. Example:

```
a = 3;      // a is assigned "3"
b = 4.7;    // b is assigned "4.7"
c = 'A';    // c is assigned 'A'
```

Rules for naming variables in C

1) First character must be an alphabet or underscore (_)
2) No commas, blanks allowed.
3) No special symbol other than (_) allowed.
4) Variable names are case sensitive.

We must create meaningful variable names in our programs. This enhances readability of our programs.

Constants

An entity whose value doesn't change is called as a constant.

A variable is an entity whose value can be changed.

URBAN EDGE

Types of constants

Primarily, there are three types of constants:

1) Integer Constant → -1, 6, 7, 9
2) Real Constant → -322.1, 2.5, 7.0
3) Character Constant → 'a', '$', '@' (Must be enclosed within single inverted commas)

Keywords

These are reserved words, whose meaning is already known to the compiler. There are 32 keywords available in C.

auto	double	int	struct
break	long	else	switch
case	return	enum	typedef
char	register	extern	union
const	short	float	unsigned
continue	signed	for	void
default	sizeof	goto	volatile
do	static	if	while

Our First C Program

```
#include <stdio.h>

int main() {
printf("Hello, I am learning C with Harry");
return 0;
}
```

File: first.C

URBAN EDGE ___/___/___

Basic Structure of a C Program

All C programs have to follow a basic structure.
A C program starts with a main function and executes instructions present inside it.
Each instruction is terminated with a semicolon (;)

There are some rules which are applicable to all the C programs :

1. Every program's execution starts from main() function.
2. All the statements are terminated with a semicolon.
3. Instructions are case-sensitive
4. Instructions are executed in the same order in which they are written.

Comments

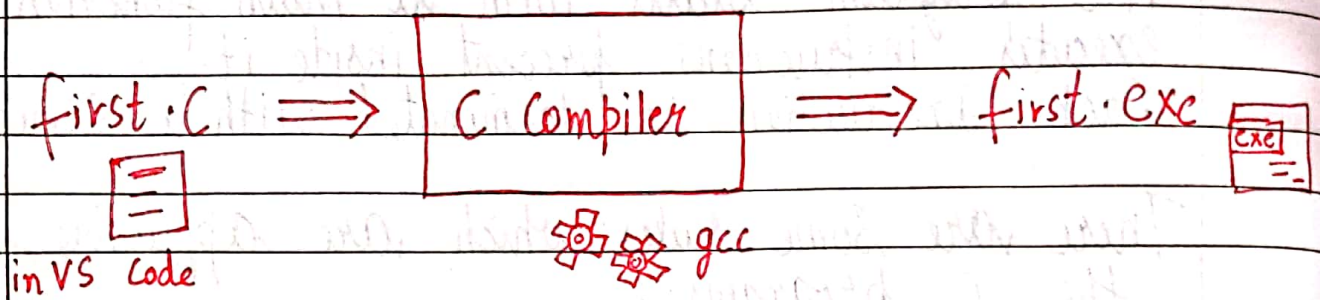
Comments are used to clarify something about the program in plain language. It is a way for us to add notes to our program. There are two types of comments in C.

1. Single line comment : // This is a comment
2. Multi-line comment : /* This is a
multi line comment */

Comments in a C program are not executed and are ignored.

URBAN EDGE

Compilation and Execution



A compiler is a computer program which converts a C program into machine language so that it can be easily understood by the computer.

A C program is written in plain text. This plain text is combination of Instructions in a particular sequence. The compiler performs some basic checks and finally converts the program into an executable.

Library Functions

C language has a lot of valuable library functions which is used to carry out certain tasks. for instance printf function is used to print values on the screen

```
printf("This is %d", i);
```

%d for integers
%f for real values
%c for characters

URBAN EDGE

Types of variables

1. Integer variables → int a = 3;
2. Real variables → int a = 7.7; float a = 7.7; ← Wrong as 7.7 is real
3. Character variables → char a = 'B';

Receiving input from the User

In order to take input from the user and assign it to a variable, we use Scanf function

Syntax for using scanf :

```
Scanf ("%d", &i);
```

↳ This & is important!

& is the "address of" operator and it means that the supplied value should be copied to the address which is indicated by variable i.

URBAN EDGE

Chapter 1 - Practice Set

Q1 Write a C program to calculate area of a rectangle:

(a) Using hard coded inputs
(b) Using inputs supplied by the User

Q2 Calculate the area of a circle and modify the same program to calculate the volume of a cylinder given its radius and height.

Q3 Write a program to convert Celcius (Centigrade degrees temperature to Farenheit)

Q4 Write a program to calculate simple interest for a set of values representing principal, no of years and rate of interest.

URBAN EDGE

Chapter 2: Instructions and Operators

A C program is a set of Instructions. Just like a recepie - which contains Instructions to prepare a particular Dish.

Types of Instructions

1. Type declaration Instruction
2. Arithmetic Instruction
3. Control Instruction

Type declaration Instruction

```
int a;
float b;
```

Other Variations:

```
int i=10; int j=i; int a=2
int j1= a + j - i;
```

```
float b = a+3; float a=1.1
```

=> ERROR! as we are trying to use a before defining it.

```
int a, b, c, d;
a = b = c = d = 30;
```

=> Value of a, b, c & d will be 30 each.