C-Programming

Introduction

C is a structural programming language developed by Dennis Ritchie at Bell Telephone Laboratories in 1972.

> C is a structural language as it can be divided into small logical and functional modules.

Features of C

- It has features of low level and high level programming language.
- It is procedural language. (i.e. structural language)
- It is fast, portable and available on all platform.

> C is widely used as it is general purpose, fast and its programs are small and portable.

> C-Programs are compiled. Compiler converts source code to object code in machine readable format.

C VS Q-Basic

- Both are:
    - High level
    - Structural
- Difference:
    - QBASIC doesn't support low level programming whereas C does.
    - QBASIC can't be used to develop system software but C can.
    - QBASIC has limited data types compared to C.
    - QBASIC has both `sub` and `function` procedure but C only has function.

Elements of C

- Character set is a set of characters that represent information in C language.

```
(+ - * & ^ % $ @ # ! ? < > { } [ ] ( ) | / \ " ' : ; . , _ )
```

- Keywords are special words which has predefined meaning in C. Keywords are reserved words.

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

- Identifiers are names given tto programming entities or elements. Rules:
    - Can contain letters, digits and underscore
    - First character must be underscore or letter
    - Keywords cannot be used as identifiers
    - Can be upto 31 characters long
    - Case sensitive
- Data types determine the type and size of data(values) associated with variables. There are four basic data types:

Data Type	Description	Size (in bytes)	Range	Format Specifier
`char`	Character	1	-128 to 127 or 0 to 255	`%c`
`int`	Integer	2 or 4	-32,768 to 32,767 or -2,147,483,648 to 2,147,483,647	`%d`
`float`	Floating point	4	1.2E-38 to 3.4E+38	`%f`
`double`	Double precision floating point	8	2.3E-308 to 1.7E+308	`%lf`

> C also has derived data types like short int, long int, unsinged char, signed char etc. `signed`, `unsigned`, `long` and `short` are called data type modifiers.

- Variables are entities whose value can be altered/changed through the program. It can be declared like this:

```
// <variable type> <name>;

int count;
int count, number;
char str;
float percentage;
```

Header files in C

> Header file is a standard file that contains function definitions, data type definitions, and macros which are necessary for functioning of a program.

- Some header files are:

Header File	Description

Header File	Description
`stdio.h`	Standard Input Output header file. It contains functions for input and output operations like `printf()` and `scanf()`.
`conio.h`	Console Input Output header file. It contains functions for console input/output like `getch()` and `clrscr()`.
`string.h`	String handling header file. It contains functions for manipulating arrays of characters (strings) like `strlen()`, `strcpy()`, and `strcat()`.
`math.h`	Mathematics header file. It contains functions for mathematical operations like `sqrt()`, `pow()`, and `sin()`.
`ctype.h`	Character type header file. It contains functions for testing and mapping characters like `isalpha()`, `isdigit()`, and `toupper()`.
`time.h`	Time header file. It contains functions for manipulating date and time like `time()`, `clock()`, and `difftime()`.

> Library functions are readymade functions in C that are defined in header files.

Comments in C

- Comments or remarks in C language provide information/description about lines of code.
- Single line comments:

```c
int a = 10; // This is  a single line comment
```

- Multiline comments

```c
/*
This is
a multi-line
comment
*/
```

Escape Characters

> Characters preceded by / and has special meaning to the compiler.

Escape Sequence	Meaning
`\n`	Newline
`\b`	Backspace
`\f`	Form feed

Escape Sequence	Meaning
\r	Carriage return
\t	Horizontal tab

Operators

> Symbols that perform some operation(calculation).

- Arithmetic Operators

Operator	Description
+	Addition
-	Subtraction
/	Division
*	Multiplication
%	Modulus (Remainder)
++	Increment
--	Decrement

- Relational Operator

Operator	Description
==	Equal to
>	Greater than
<	Less than
!=	Not equal to
>=	Greater than or equal to
<=	Less than or equal to

- Logical Operator

Operator	Description
&&	Logical AND
||	Logical OR
!	Logical NOT

- Assignment Operator

Operator	Description
`=`	Assign
`+=`	Add and assign
`-=`	Subtract and assign
`*=`	Multiply and assign
`/=`	Divide and assign
`%=`	Modulus and assign

Basic Input/Output

- You need `stdio.h` {standard Input Output} for printf() and scanf() and `conio.h` {console input output} for getch()

```
#include <stdio.h>
#include <conio.h>
```

- All of your code except those should be inside main function.
- `printf()` for output
- Syntax:

```
printf("format specifier", argument_list);
```

```
#include <stdio.h>

void main(){
    printf("Hello World");
}
```

- Syntax:

```
scanf("format string", argument_list);
```

- `scanf()` for input

```
#include <stdio.h>
```

```
void main(){
    int num;

    scanf("%d", &num); // This takes input. Notice there is '&' before num.
    printf("%d", num); // This displays output

    // You should use proper format specifier(%d, %f...)
}
```

Conditional statements

- `if...else if...else` lets you execute different statements based on conditions
- Syntax:

```
if <condition_here>:
    // block of code
else if <condition_here>:
    //block of code
//more else if blocks
else:
    //default block of code
```

- Program to get input and print if it is odd or even.

```
#include <stdio.h>
#include <conio.h>

int main() {
    int number;

    // Prompt the user to enter a number
    printf("Enter an integer: ");

    // Read the input number
    scanf("%d", &number);

    // Check if the number is odd or even
    if (number % 2 == 0) {
        printf("%d is even.\n", number);
    } else {
        printf("%d is odd.\n", number);
    }
    getch(); //This waits for keypress on keyboard. i.e. holds output until
keypress.
    return 0;
}
```

For loop

- `for` loop lets you execute a block of code a certain number of times.
- Syntax:

```
for (initialization; condition; increment/decrement) {
    // block of code
}
```

- Example program to print odd numbers from 1 to 20.

```
#include <stdio.h>
#include <conio.h>
int main() {
    int i;

    // Loop from 1 to 10
    for (i = 1; i < 20; i=i+2) {
        printf("%d\n", i);
    }

    getch();
    return 0;
}
```

- Explanation:
    - `initialization`: This is executed once at the beginning of the loop. It is typically used to initialize the loop counter variable.
    - `condition`: This is evaluated before each iteration of the loop. If it evaluates to true, the loop body is executed. If it evaluates to false, the loop terminates.
    - `increment/decrement`: This is executed after each iteration of the loop. It is typically used to update the loop counter variable.

While loop

> While loop executes a statement or block of statements till the given condition is True.

- Syntax:

```
while { condition } {
    // while loop body
}
```

- Example program to print odd numbers from 1 to 20.

```c
#include <stdio.h>
#include <conio.h>
int main() {
    int i;

    i = 1
    // Loop while i is less than 20. Then, stop.
    while(i < 20){
        printf("%d\n", i);
        i = i + 2;
    }

    getch();
    return 0;
}
```
