APJ ABDUL KALAM TECHNOLOGICAL UNIVERSITY

STUDY MATERIALS

MODULE 1

Machine language

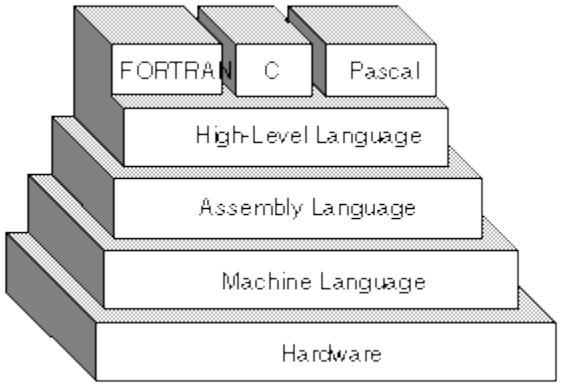
Machine language is the lowest-level programming language (except for computers that utilize programmable microcode). Machine languages are the only languages understood by computers. Machine code or machine language is a set of instructions executed directly by a computer's central processing unit (CPU). Each instruction performs a very specific task, such as a load, a jump, or an ALU operation on a unit of data in a CPU register or memory. Every program directly executed by a CPU is made up of a series of such instructions.

Assembly language

An assembly language is a low-level programming language for microprocessors and other programmable devices. It is not just a single language, but rather a group of languages. Assembly language implements a symbolic representation of the machine code needed to program a given CPU architecture.

High level language

A high-level language is a programming language such as C, FORTRAN, or Pascal that enables a programmer to write programs that are more or less independent of a particular type of computer. Such languages are considered high-level because they are closer to human languages and further from machine languages.



Compiler

A compiler is a computer program (or a set of programs) that transforms source code written in a programming language (the source language) into another computer language (the target language), with the latter often having a binary form known as object code. The most common reason for converting source code is to create an executable program.

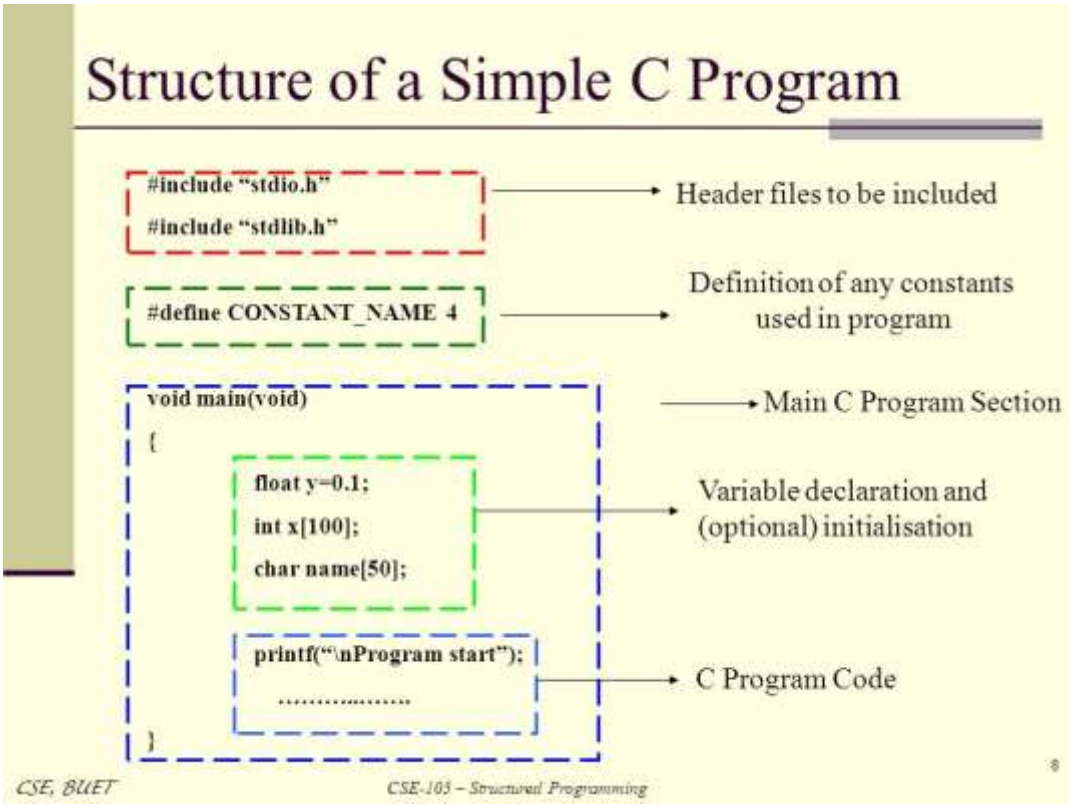
Assembler

An assembler is a program that takes assembly language and converts them into machine language that the computer's processor can use to perform its basic operations. ie, Conversion from assembly language to target language.

Flowcharts and Algorithms

Refer Note:

Structure of C Program



Keywords in C Programming Language :

Keywords are those words whose meaning is already defined by Compiler

- Cannot be used as Variable Name
- There are 32 Keywords in C
- C Keywords are also called as Reserved words .
- 32 Keywords in C Programming Language

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

Identifiers

In C language identifiers are the names given to variables, constants, functions and user-define data. These identifier are defined against a set of rules.

Rules for an Identifier

- An Indetifier can only have alphanumeric characters(a-z , A-Z , 0-9) and underscore(_).
- The first character of an identifier can only contain alphabet(a-z , A-Z) or underscore (_).
- Identifiers are also case sensitive in C. For example name and Name are two different identifier in C.
- Keywords are not allowed to be used as Identifiers.
- No special characters, such as semicolon, period, whitespaces, slash or comma are permitted to be used in or as Identifier.

Data types in C Language

Data types specify how we enter data into our programs and what type of data we enter. C language has some predefined set of data types to handle various kinds of data that we use in our program. These datatypes have different storage capacities.

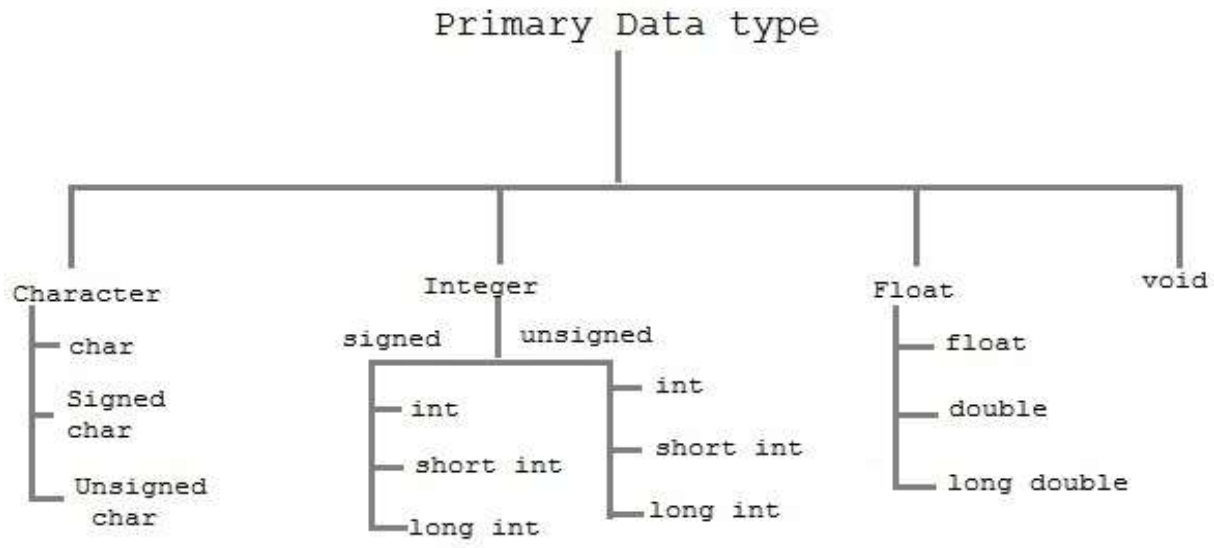
C language supports 2 different type of data types,

1. Primary data types

These are fundamental data types in C namely integer(int), floating(float), charater(char) and void.

2. Derived data types

Derived data types are like arrays, functions, stuctures and pointers.



integer type

Integers are used to store whole numbers.

Size and range of Integer type on 16-bit machine

Type	Size(bytes)	Range
int or signed int	2	-32,768 to 32767
unsigned int	2	0 to 65535
short int or signed short int	1	-128 to 127
long int or signed long int	4	-2,147,483,648 to 2,147,483,647
unsigned long int	4	0 to 4,294,967,295

Floating type

Floating types are used to store real numbers.

Size and range of Integer type on 16-bit machine

Type	Size(bytes)	Range
Float	4	3.4E-38 to 3.4E+38
Double	8	1.7E-308 to 1.7E+308
long double	10	3.4E-4932 to 1.1E+4932

Character type

Character types are used to store characters value.

Size and range of Integer type on 16-bit machine

Type	Size(bytes)	Range
char or signed char	1	-128 to 127
unsigned char	1	0 to 255

void type

void type means no value. This is usually used to specify the type of functions.

Input Output Functions in C

The getchar() and putchar() Functions

The int getchar(void) function reads the next available character from the screen and returns it as an integer. This function reads only single character at a time. You can use this method in the loop in case you want to read more than one character from the screen.

The int putchar(int c) function puts the passed character on the screen and returns the same character. This function puts only single character at a time. You can use this method in the loop in case you want to display more than one character on the screen. Check the following example –

```
#include <stdio.h>
main( )
 {
   int c;
   printf( "Enter a value :");
   c = getchar( );
   printf( "\nYou entered: ");
   putchar( c );

}
```

When the above code is compiled and executed, it waits for you to input some text. When you enter a text and press enter, then the program proceeds and reads only a single character and displays it as follows −

```
$./a.out
Enter a value : this is test
You entered: t
```

The gets() and puts() Functions

The char *gets(char *s) function reads a line from stdin into the buffer pointed to by s until either a terminating newline or EOF (End of File).
The int puts(const char *s) function writes the string 's' and 'a' trailing newline to stdout.

```
#include <stdio.h>
main( )
{
   char str[100];
   printf( "Enter a value :");
   gets( str );
   printf( "\nYou entered: ");
   puts( str );

}
```

When the above code is compiled and executed, it waits for you to input some text. When you enter a text and press enter, then the program proceeds and reads the complete line till end, and displays it as follows −

```
$./a.out
Enter a value : this is test
You entered: this is test
```

The scanf() and printf() Functions

The int scanf(const char *format, ...) function reads the input from the standard input stream stdin and scans that input according to the format provided.

The int printf(const char *format, ...) function writes the output to the standard output stream stdout and produces the output according to the format provided.

The format can be a simple constant string, but you can specify %s, %d, %c, %f, etc., to print or read strings, integer, character or float respectively. There are many other formatting options available which can be used based on requirements. Let us now proceed with a simple example to understand the concepts better −

```
#include <stdio.h>
main( )
 {
   char str[100];
   int i;
   printf( "Enter a value :");
   scanf("%s %d", str, &i);
   printf( "\nYou entered: %s %d ", str, i);


}
```

When the above code is compiled and executed, it waits for you to input some text. When you enter a text and press enter, then program proceeds and reads the input and displays it as follows −

```
$./a.out
Enter a value : seven 7
You entered: seven 7
```

Here, it should be noted that scanf() expects input in the same format as you provided %s and %d, which means you have to provide valid inputs like "string integer". If you provide "string string" or "integer integer", then it will be assumed as wrong input. Secondly, while reading a string, scanf() stops reading as soon as it encounters a space, so "this is test" are three strings for scanf().

try it now

A KTU STUDENTS PLATFORM

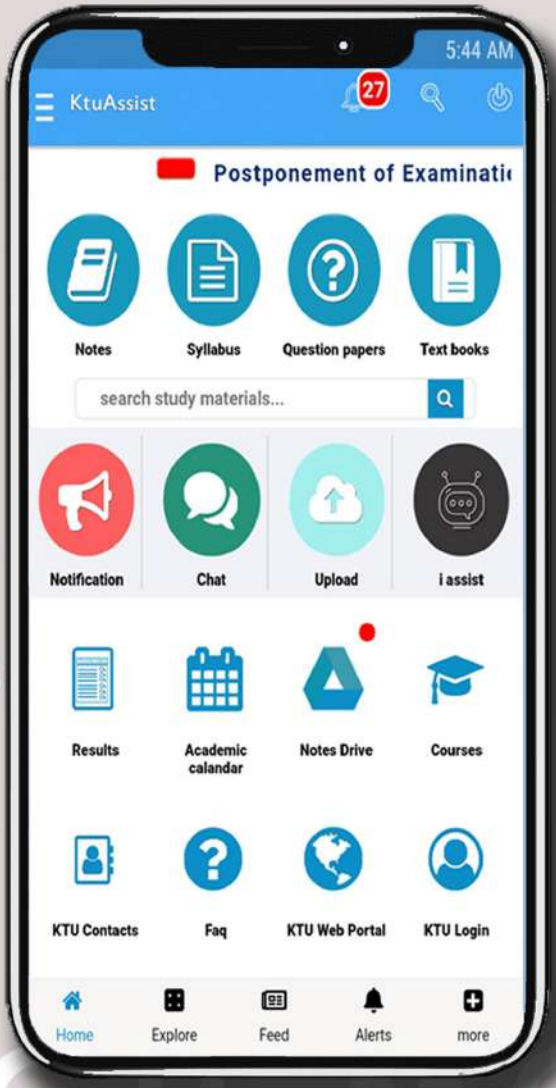
SYLLABUS

NOTES

TEXT BOOKS

QUESTION PAPERS

KTU NOTIFICATION



DOWNLOAD IT FROM GOOGLE PLAY

CHAT

FAQ

LOGIN

CALENDAR

MUCH MORE

DOWNLOAD APP

ktuassist.in

instagram.com/ktu_assist

facebook.com/ktuassist