| Introduction to Programming  **syllabu**s: Machine language, assembly language, and high level language. Compilers and assemblers. Flow chart and algorithm – Development of algorithms for simple problems. Basic elements of C: Structure of C program –Keywords, Identifiers, data types, Operators and expressions – Input and Output functions. |
| --- |



**Variables**

Variables are the symbolic name given to a particular portion of memory in which the data need to be processed is stored.(ie,variables are using to store values).

Variable data can be manipulated using instructions during execution of program .but a variable can store only one value at a time .in order to use a variable firstly we have declare it.

A variable can be declared using the following syntax

**Syntax**

**Data type variable name;**

Data type is the type of data need to be stored in the variable.we can provide name to the variable based on our need.but we have to consider the following rules.

**Rules of naming variable**

* Key words can’t be use as variable
* variables are case sensitive in c,ie upper case and lower case letters are different for c.

Eg:

Consider two statements

**Int max; (lower case)**

**int MAX; (upper case)**

Here “max” and “MAX” are different variables

* Spacing are not allowed in variable names.

Ie, we can write **Int maxmin;** but not **Int max min;**

**Constants**

Constant is a value that can be stored in memory and use for processing .but it cannot be manipulated using instructions during execution of program.

1. **Numeric constants**
2. **Integer constants : eg:4,5 etc**
3. **Floating point constants: eg:3.4,5.6 etc**
4. **Character constants: eg:”a”, ”7” ,”$” etc**

**Keywords and identifiers**

Every word in a c program can be classified as either keywords or identifiers

**Keywords**

* Keywords are the words using in c programming which has predefined meaning in c compiler.
* A User can’t change meaning of a keyword
* There are 32 keywords in c language.some of them are

**Int,char,float,if,else,void ,struct,union,return,do,while etc**

* Since c is case sensitive all keywords should be written in lower case letters(ie,in small letters).otherwise compiler will show an error.
* For example,consider a variable declaration.

**Int x;**

Here “int” is the keyword,and it means the data using in variable “x” is integer type

**Identifiers**

* Identifiers are sequence of characters or words which are using for naming entities such as variables,constants,functions,arrays etc.
* Identifiers does not have predefined meaning.
* For Example take declaration of a variable “x”.

**Int x;**

Here “x” is the identifier and “int” is the keyword**.**

* Consider another example

**Int max[3];**

Here we are declaring an array named max.in this “max” is the identifier and int is the keyword

**There are certain rules regarding identifiers**

* An identifier should contain only alpha numeric characters(ie, a-z,A-Z,1-9)and underscore( \_ ).
* The first character of an identifier must be a alphabet(ie,a-z,A-Z) or underscore not a number.
* Identifiers must not use any keywords (ie, “ **int int;** ” is a wrong statement)
* Identifiers are also case sensitive in c,ie upper case and lower case letters are different for c.

Eg:

Consider two statements

**Int max; (lower case)**

**int MAX; (upper case)**

Here “max” and “MAX” are different identifiers

* Special characters like comma,semicolon,slash cannot be use in a identifier

**Operators**

Operators are the symbols that tell the computer to perform certain mathematical or logical operations.

The data or variables need to be manipulated using Operators is called as **operands**.

Eg:

Consider an expression, X+Y;

Here X and Y are the variables that contain data need to be manipulated.so X and Y are our operands.

The symbol "+" is called as operator.this symbol tells the computer to perform addition of data present in variables X and Y.

**Operators that are using in c language**

1. Arithmetic operator
2. Logical operator
3. Relational operator
4. Increment or decrement operator
5. Assignment operator
6. Conditional operator
7. Bit wise operator
8. Special operator

**Arithmetic operator**

Arithmetic operators are the symbols that tells the computer to perform arithmetical operations.

| **Operator** | **meaning** |
| --- | --- |
| **+** | addition |
| **\_** | subtraction |
| **\*** | multiplication |
| **/** | division |
| **%** | Modular division |

Eg:

Assume A=3 and B=2

* **A+B=5**
* **A - B=1**
* **A\*B=6**
* **A/B=1** (this operator will avoid reminder in output)
* **A%B=5**  (this operator will take only reminder in output)

**Logical operator**

Symbols that tells the computer to perform logical operations(like AND,OR, etc)

| operators | meaning | Example(assume A=1,C=0) |
| --- | --- | --- |
| && | Logical AND (If both operands are non zero,then output will be true) | A&&C  Output will be false |
| **||** | Logical OR (output will be true If any one of the operand is one) | A**||**C  Output will be true |
| **!** | Logical NOT (using to reverse the output) | **!**(A&&C)  Output will be true |

**Rotational operator**

These are symbols that tells the computer to perform comparison of variable data.

| **operator** | **meaning** |
| --- | --- |
| = = | Equal to |
| **!**= | Not equal to |
| < | Less than |
| < = | Less than or equal to |
| > | Greater than |
| > = | Greater than or equal to |

**Increment and decrement operator**

Using to increment or decrement variable data by one.

| **operator** | **meaning** |
| --- | --- |
| **+ +** | increment operator |
| \_ \_ | decrement operator |

Increment and decrement operators can be use in two types

1. Pre Increment or decrement
2. Post Increment and decrement

Eg:-

assume a=5

* **b=a++** (post increment operation ,the value 5 is assigned to ‘b’, and then ‘a’ is incremented to 6)
* **b=++a** (pre increment operation ,The value of ‘a’ is incremented to 6 and then 6 is assigned to ‘b’)
* **b=a - -** (post decrement operation ,the value 5 is assigned to ‘b’, and then ’a’ is decrement to 4)
* **b= - -a**  (pre decrement operation,The value of ‘a’ is decremented to 4 and then 4 is assigned to’ b’)

**Assignment operator**

Assignment operator are the symbols that are using to assign values to the variables

.

| operator | example |
| --- | --- |
| = | C=2 (assign 2 to variable C)  C=a+b (assign value of a**+**b to variable C) |
| += | C**+=**2 (assign value of C**+**2 to variable C)  A+=3 (assign value of A**+**3 to variable A) |
| **-** = | C **-** =2 (assign value of C**-** 2 to variable C) |
| /= | C**/=**2 (assign value of C**/**2 to variable C) |
| \*= | C\*=2 (assign value of C\*2 to variable C) |
| >>= | C>>=2(doubt) |
| <<= | C<<=2(doubt) |
| &= | C&=2(doubt) |

**Conditional operators**

**Syntax:**

**expression 1 ? expression 2 : expression 3**

The operators works as follows,expression 1 is evaluated first.

If expression 1 is true, then expression 2 value will be returned.

If expression 1 is false, then expression 3 value will be returned.

Eg:

Consider **Y**= **(x<5 ? 7 : 8);**

if x=4, expression 1 become true and y =7

If x=6,expression 1 become false and y =8

**Bit wise operators**

Bit wise operators are used for the manipulation of data at bit level. These operators are used for testing the bits, or shifting them right or left.

| operator | meaning |
| --- | --- |
| **&** | Bit wise AND |
| **|** | Bit wise OR |
| **<<** | Left shift |
| **>>** | Right shift |
| **^** | Bit wise XOR |

**(Refer tp to know more)**

**Special operators**

Operators using for special operations**.**

1. **Comma operator ( “, “)**

This operator is Using to link related expressions together.

Eg:

Consider the expression **A=(X=5,Y=4,X+Y);**

In this expression,firstly 5 is assigned to X and 4 is assigned to Y. then operation addition of X and Y takes place and result will assign to A

1. **Sizeof operator**

This is a compile operator which is used when size of entities like array,structure and variables are unknown to the programmer.

We can find the size of an unknown entity or variables using size of operator.

It can also be used for dynamic memory allocation.

**Syntax:**

**Sizeof (unknown entity or variable whose size need to be find);**

Eg:

**A=sizeof (x);**

This expression will assign size of variable “x” to variable “a”.

1. **pointer operator**

Pointer operators are using in when we are dealing with pointer

| **operator** | **name** | **use** |
| --- | --- | --- |
| **&** | Address operator | Using to add data address to pointer |
| **\*** | Value operator | Using to access data using pointer |

1. **Member selection operator**

| **Operator** | **name** | **use** |
| --- | --- | --- |
| **.** | Dot operator | Using to access data stored in members of structure and union |
|  | Arrow operator | Using to access data stored in members of structure and union |

**EXPRESSION**

Expression is the arrangement of variables,constants and operators as per syntax.

Some Expressions are Arithmetic expression ,Logical expression ,Relational expression etc

**Data types in c**

The Types of data that can be use in c programming language can be broadly classified into 3

1. **Basic data types or primary data type**

these are the built in data type.predefined while developing c language.

1. Integer data type
2. Character data type
3. Floating point data type
4. Double data type
5. **Derived data types**

These are Data types derived from basic data type

1. Arrays
2. Functions
3. pointers
4. **User defined data types:** Data type defined by programmer.
5. Structure
6. Union

Data type are necessary to declare a variable.data types will decides what kind of data can be store in a variable and size of memory need to be allocated for that.

1. **Basic data types**

Basic data types using in c language are

1. **Character data type**

* We can store and process character values in c language.
* Char is the key word for character data type.e if we want to store a character value in a variable we have to follow the below syntax.

**char variable name;**

**Eg:**

**Char x;**

* Size of a char is 1 byte

1. **Integer data type**

* We can store and process integers values in c language.
* Int is the keyword for integer data type.ie if we want to store an integer value in a variable we have to follow the below syntax.

**Int variable name;**

**Eg:**

**Int x;**

* Size of an integer data is 2 byte

1. **Float data type**

* We can store and process real number values(ie, values with decimal. Eg:3.5) in c language.
* float is the keyword for real number data type.ie if we want to store real number in a variable we have to follow the below syntax.

**float variable name;**

**Eg:**

**float x;**

* Size of a float data is 4 byte

1. **Double data type**

* We can store and process double precision real numbers (eg: 3.56)
* Double is the keyword for double precision real number data type.ie if we want to store double precision real number in a variable we have to follow the below syntax.

**double variable name;**

**Eg:**

**double x;**

* Size of a double data is 8 byte.

**Input output functions in c**

I/o functions in c language is using to read data from output device and send data to output device.

**Classification of input output functions**

Input output functions in c is classified into 2

1. Standard or formatted i/o functions
2. Unformatted i/o functions
3. **Standard or formatted i/o functions**

Standard or formatted i/o functions are built in functions.these are stored in ‘stdio.h’ header file.so in order to use Standard or formatted i/o functions in a program we have to link our program with stdio.h file.

Using Standard or formatted i/o functions we can read and write all type of data.

After the execution of instruction Standard or formatted i/o functions always return values which is equal to the no.of variables read or write.this can be use to check possibility of error in execution.

1. **Standard input function**

**Scanf()** is the standard input function in c.this function is using to read data from output devices

**Syntax:**

**Scanf(“control string”,&variable 1,&variable2 …..);**

**Eg:**

**Scanf(“%d%f%C”,&x,&y,&z);**

**x,y,z variables are integer,float and character types respectively**

1. **Standard output function**

**printf()** is the standard output function in c.this function is using to send data to output devices

**Syntax:**

**printf(“control string”,variable 1,variable2 …..);**

**Eg:**

**printf(“%d%f%C”,x,y,z);**

**x,y,z variables are integer,float and character types respectively**

1. **Unformatted i/o functions**

This kind of functions can handle character datatype and string values.some of unformatted functions are

* **Getchar() :** this is an unformatted input function which is using to read character data from output device.

**Syntax:**

**Variable=getchar();**

**Eg:**

**A=getchar();**

* **Putchar() :**this is an unformatted output function which is using to send character data to output device.

**Syntax:**

**putchar(variable);**

**Eg:**

**putchar(a);**

* **Getch() :** this is an unformatted input function which is using to read character data from output device.

**Syntax:**

**Variable=getch();**

**Eg:**

**A=getch();**

* **Putch() :**this is an unformatted output function which is using to send character data to output device.

**Syntax:**

**putch(variable);**

**Eg:**

**putch(a);**

* **Gets() :**this is an unformatted input function which is using to read string data from output device.

**Syntax:**

**Gets(string name);**

* **Puts() :**this is an unformatted output function which is using to send string data to output device.

**Syntax:**

**puts(string name);**

**M1**

* **Program to swap 2 variable data without using a 3rd variable**

**#include <stdio.h>**

**#include <conio.h>**

**void main()**

**{**

**clrscr();**

**Int x,y;**

**printf(“enter numbers\n”);**

**Scanf(“%d%d”,&x,&y);**

**x=x+y;**

**y=x-y;**

**x=x-y;**

**printf(“%d %d\n”,x,y);**

**getch();**

**}**

* **Program to find area of triangle using value of 3 sides**

**#include <stdio.h>**

**#include <conio.h>**

**Void main()**

**{**

**Clrscr();**

**Int x,y,z,area;**

**Printf(“enter length of sides\n”);**

**Scanf(“%d%d%d”,&x,&y,&z);**

**S=(x+y+z) /3;**

**Area=root of (s(s-x)(s-y)(s-z));**

**Printf(“%d\n”,area);**

**getch();**

**}**