# Keywords, Identifier, Literals, Operators and Expression Assignment

**Mandatory:**

1. **Choose all valid identifiers**
   1. **int int**
   2. **int \_numvalue**
   3. **float price\_money**
   4. **charname1234567890123456789012345678901234567890**
   5. **char name value**
   6. **char $name**

Ans: b and c are valid identifiers.

1. **What is the meaning of the following keywords, show the usage**
   1. **Auto**

Ans: The auto keyword **declares automatic variables**. For example: auto int var1; This statement suggests that var1 is a variable of storage class auto and type int. Variables declared within function bodies are automatic by default.

* 1. **Extern**

Ans: In C, the 'extern' keyword is **used to declare global variables**. By default, functions are global. Hence, the 'extern' keyword does not make any changes in functions.

* 1. **Volatile**

Ans: A volatile keyword in C is nothing but a qualifier that is used by the programmer when they declare a variable in source code. It is used to inform the compiler that the variable value can be changed any time without any task given by the source code. Volatile is usually applied to a variable when we are declaring it.

* 1. **Sizeof**

Ans: The **sizeof**() function in **C** is a built-in function that is used to calculate the size (in bytes)that a data type occupies in ​the computer's memory.

* 1. **Const**

Ans: We **use the const qualifier to declare a variable as constant**. That means that we cannot change the value once the variable has been initialized. **allows a programmer to tell the compiler that a particular variable should not be modified after the initial assignment in its declaration**.

1. **Explain the difference between the following variables.**
   1. **char \*ptr = “ABC”;**
   2. **char arr[]=”ABC”;**

Ans: The statements ‘**char arr[] = “ABC”**‘ creates a character array which is like any other array and we can do all array operations. The only special thing about this array is, although we have initialized it with 9 elements, its size is 10

The statement ‘**char \*ptr = “ABC”**‘ creates a string literal. The string literal is stored in the read-only part of memory by most of the compilers. The C and C++ standards say that string literals have static storage duration, any attempt at modifying them gives undefined behaviour.   
**ptr** is just a pointer and like any other pointer stores address of string literal.

**Can you manipulate the contents of ptr? Why?**

Ans: An element is accessed by indexing the array name. This is done by placing the index of the element within square brackets after the name of the array.

**Can you manipulate the contents of arr? Why?**

Ans: The most convenient way of performing array manipulation is to use the for repetitive construct (for loop) for accessing elements of the array.

**Which one of the above is a string literal?**

Ans: char \*ptr = “ABC”; is a string literal.

1. **Predict the output of the following code .**

**void main()**

**{**

**//set a and b both equal to 5.**

**int a=5, b=5;**

**//Print them and decrementing each time.**

**//Use postfix mode for a and prefix mode for b.**

**printf("\n%d %d",a--,--b);**

**printf("\n%d %d",b++,--b);**

**}**

Ans: 5 4

3 4

1. **Refer the code snippet. It fails with error. Fix it.**

**#include<stdio.h>**

**int main()**

**{**

**int i,k;**

**const int num;**

**/\* for(i = 0;i < 9;i++)**

**{**

**k = k + 1;**

**} \*/**

**num = num + k; /\* Compiler gives the error here \*/**

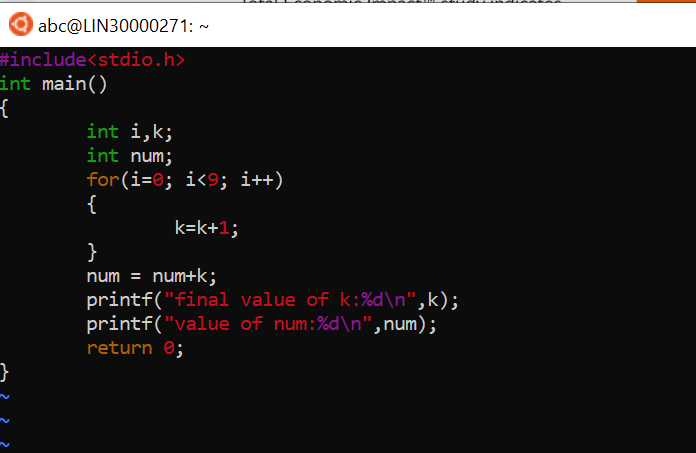
**printf("final value of k:%d\n",k);**

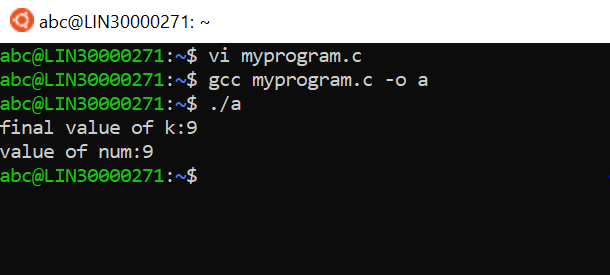
**printf("value of num:%d\n",num);**

**return 0;**

**}**

Ans: On running above code, it will give an error because of declaration on “const” keyword. So on removing it and removing comments we will get output as:





**6.Consider the following code snippet. Evaluate the value of f1, f2 and f3.**

**int main()**

**{**

**int i = 10;**

**int j = 3;**

**float f1 = i / j;**

**float f2 = (float ) i / j;**

**float f3 = (float ) (i / j);**

**}**

Ans: 