**Keywords, Identifier, Literals, Operators and Expression Assignment**

Mandatory:

1. Choose all valid identifiers

a. int int

**b. int \_numvalue**

**c. float price\_money**

**d. char name1234567890123456789012345678901234567890**

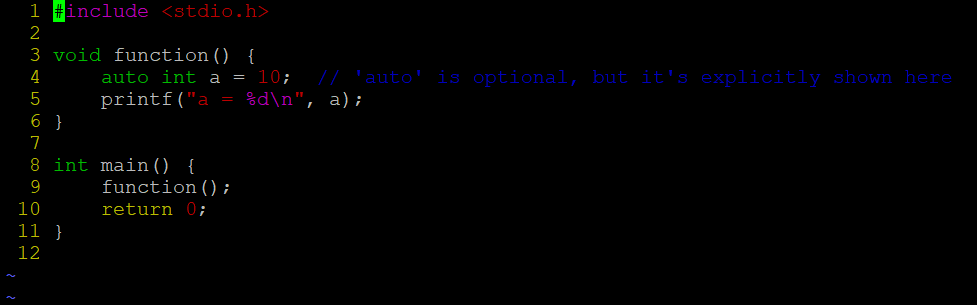
e. char name value

f. char $name

2. What is the meaning of the following keywords, show the usage

**a. auto :**

* **auto is a storage class specifier in C, which indicates that a variable has automatic storage duration. This means that the variable is automatically created when the block (usually a function) is entered and destroyed when the block is exited.**
* **auto cannot be used for global or static variables.**



**b. extern :**

* **extern is used to declare a variable or function that is defined in another file or elsewhere in the program.**
* **It tells the compiler that the variable or function is defined elsewhere, but its memory allocation or definition will be resolved during the linking phase.**
* **It is commonly used for global variables and functions that need to be accessed across different files.**

A screenshot of a computer screen

Description automatically generated

**c. volatile :**

* **volatile is a keyword that tells the compiler that a variable's value may be changed by external factors (like hardware or another thread) and that the compiler should not optimize or cache it.**
* **It is commonly used for hardware registers, shared memory locations in multi-threaded programs, or flags that may change unexpectedly.**

A screenshot of a computer

Description automatically generated

**d. sizeof :**

* **sizeof is an operator, not a keyword, that returns the size (in bytes) of a data type or variable.**
* **It is commonly used to determine the size of data types and structures at compile time.**

A screen shot of a computer

Description automatically generated

**e. const :**

* **const is used to declare that a variable's value is constant and cannot be changed after initialization.**
* **It can be applied to any data type and is commonly used for constants, especially when the value should not be modified.**

A screenshot of a computer

Description automatically generated

3. Explain the difference between the following variables.

a. char \*ptr = “ABC”;

**char \*ptr = "ABC";**

* **It declares a pointer ptr that points to a string literal "ABC".**
* **In C, string literals (like "ABC") are stored in a read-only section of memory, and ptr is a pointer to the first character of this string literal.**
* **The string "ABC" is automatically null-terminated, meaning it is stored as an array of characters: {'A', 'B', 'C', '\0'}.**

b. char arr[]=”ABC”;

**char arr[] = "ABC";**

* **It declares an array of characters (arr) and initializes it with the string literal "ABC". The array is large enough to hold all the characters in the string, including the null terminator ('\0'), so arr will have the following content: {'A', 'B', 'C', '\0'}.**
* **This array is stored in writable memory (not read-only), and the contents of the array can be modified.**

Can you manipulate the contents of ptr? Why?

**No: We cannot manipulate the contents of ptr. Since the string literal "ABC" is typically stored in a read-only section of memory (depending on the compiler and platform), attempting to modify the contents through ptr would result in undefined behavior (most likely a segmentation fault or access violation).**

Can you manipulate the contents of arr? Why?

**Yes: We can manipulate the contents of arr. Since arr is a character array and not a pointer to a string literal, it is stored in writable memory, meaning you can change the individual characters of the array.**

Which one of the above is a string literal?

**In the case of char \*ptr = "ABC";, "ABC" is a string literal because it is a constant string stored in a read-only section of memory.**

**In char arr[] = "ABC";, "ABC" is a string literal, but the array arr itself is not a string literal — it is an array that copies the string literal into writable memory.**

4. 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);

}

A screen shot of a computer

Description automatically generated

5. 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;

}

A screenshot of a computer program

Description automatically generated

A screen shot of a computer

Description automatically generated

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; 🡺 **3.00**

float f2 = (float ) i / j; 🡺 **3.33**

float f3 = (float ) (i / j); 🡺 **3.00**