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

A)int\_numvalue, float price\_money, char name1234567890123456789012345678901234567890, char $name are valid identifiers.

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

a. auto : auto is used to automatically deduce the type of a variable based on its initializer (in C++11 and later). It is no longer widely used in C, but in C++, it is used to infer the type.

b. extern : extern is used to declare a variable or function that is defined elsewhere (typically in another source file).

c. volatile : volatile is used to tell the compiler not to optimize a variable, as its value may be changed by external factors, such as hardware or a different thread.

d. sizeof : sizeof is used to determine the size, in bytes, of a data type or variable.

e. const : const is used to declare variables whose values cannot be changed after initialization.

3. Explain the difference between the following variables.

a. char \*ptr = “ABC”;

ptr is a pointer to a string literal. In C, string literals are stored in read-only memory (depending on the implementation). The pointer ptr points to the first character of the string "ABC".

b. char arr[]=”ABC”;

arr is an array of characters. It is initialized with the string "ABC". The array arr contains the characters {'A', 'B', 'C', '\0'}.

Can you manipulate the contents of ptr? Why?

No, because the string literal "ABC" is stored in a read-only section of memory (by most compilers), so modifying it results in undefined behavior

Can you manipulate the contents of arr? Why?

Yes, you can modify the contents of arr because arr is an actual array with writable memory

Which one of the above is a string literal?

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);

}

Output:

5 4

4 4

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;

}

Output: You can't modify num after it has been declared as const. So, either remove the line num = num + k; or avoid modifying num.

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

intmain()

{

int i = 10;

int j = 3;

float f1 = i / j;

float f2 = (float ) i / j;

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

}

Output: f1 = 3.0

f2 = 3.333...

f3 = 3.0