Keywords, Identifier, Literals, Operators and Expression Assignment

Mandatory:

1. Choose all valid identifiers
    a. int int

    Not valid. "int" is a keyword and cannot be used as an identifier.

    b. int _numvalue

    Valid. It starts with an underscore, which is allowed in C, and follows proper identifier naming rules.

    c. float price_money

    Valid. This is a correct identifier that follows the rules.

    d. char name12345678901234567890123456789012345678 90

    Valid, but impractical. Although it is a valid identifier because it only contains letters, digits, and underscores, it is far too long and may cause readability or other issues.

    e. char name value

    Not valid. Identifiers cannot contain spaces.

    f. char $name

    Valid. In C, the dollar sign ($) is a valid character in identifiers, though it is rarely used.

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

    Meaning: auto is used to define automatic variables in C. By default, local variables are automatically auto. This keyword is rarely used nowadays since the default behaviour is the same.

    Ex: auto int x=5;

    b. Extern

    Meaning: extern is used to declare a variable or function that is defined in another file. It allows access to variables/functions across different files.

    EX: extern int num;

    c. Volatile

Meaning: volatile tells the compiler that a variable can be changed unexpectedly, e.g., by hardware or a different thread. The compiler won't optimize the use of that variable.

EX: volatile int counter;

d. Sizeof

Meaning: sizeof is an operator used to determine the size (in bytes) of a data type or variable.

printf("Size of int: %zu", sizeof(int));

e. Const

Const is used to declare a constant variable whose value cannot be modified after initialization.

Ex: const int max_value = 100;

3. Explain the difference between the following variables.
   a. char *ptr = “ABC”;
   b. char arr[]=”ABC”;

   This declares a pointer ptr that points to the first character of a string literal "ABC".

   This creates a character array arr that stores the string "ABC" with a null terminator \0 at the end.

   Can you manipulate the contents of ptr? Why?
   - **You can change where ptr points, but you cannot modify the string literal i.e the contents of "ABC", as string literals are typically stored in read-only memory.**

   Can you manipulate the contents of arr? Why?
   - **Yes, you can modify the contents of the array, e.g., arr[0] = 'X';, because arr is a modifiable array in memory.**

   Which one of the above is a string literal?
   - **"ABC" 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 3

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;
}
```

```
#include<stdio.h>
int main()
{
        int i,k=0;
        const int num=10;
        /*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;
}
```

```
user72@trainux01:~$ vi prog.c
user72@trainux01:~$ gcc prog.c
user72@trainux01:~$ ./a.out
final value of k:0
value of num:10
```

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);
}
```

F1 =3.0

F2=3.3333

F3=3.0