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

Mandatory:

1. Choose all valid identifiers

**a. int int** – it is not valid identifier, because int is a reserved word

**b. int \_numvalue** – It is a valid identifier, because first identifier can be alphabet or underscore.

**c. float price\_money** – It is a valid identifier

**d. char name1234567890123456789012345678901234567890** – It is a valid identifier

**e. char name value** – it is not a valid identifier, because it has space.

**f. char $name** – It is not a valid identifier, because special characters included in it.

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

**a. auto** – it is a storage class specifier, indicates that a variable is automatic, similar to local variables. By default, all local variables in C are automatic, so don’t need to use auto keyword explicitly.

**b. extern** – used to declare a variable or function that is defined in another source file or module.

**c. volatile** – it is used to indicate that a variable can be modified by external outside the program’s control. Variable’s value is fetched directly from its memory location.

**d. sizeof** – it is used to know the size.

**e. const** – used to define constants, that cannot be modified once assigned.

**3. Explain the difference between the following variables.**

**a. char \*ptr = “ABC”;** - ptr is a pointer variable that points to a string literal “ABC”.

**b. char arr[]=”ABC”;** - arr stores the characters of the string “ABC”.

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

No, cannot modify the contents of ptr directly. Because, pointer points to starting address of “ABC”. Modifying location is not allowed and it gives segmentation fault error.

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

Yes, we can modify the contents of arr. Because, “ABC” is copied into the arrays memory location and can be modified.

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

char \*ptr = “ABC”, because it is stored in read-only 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); // a = 5, b = 4

printf("\n%d %d",b++,--b); b = 3, b = 4

}

Output:

5 4

3 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;

}

A. Here ‘num’ is assigned with ‘const int’. So, ‘num’ value(num=num+k) cannot be modified error will be assignment of read-only variable num. So to fix the error declare ‘num’ as just ‘int num’ without giving ‘const’ keyword.

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; // integer division , then result is implicitly converted to float

float f2 = (float ) i / j; // floating-point division.

float f3 = (float ) (i / j); // integer division , then result is explicitly converted to float

}

Output:

f1 = 3.000000

f2 = 3.333333

f3 = 3.000000