# Keywords, Identifier, Literals, Operators and Expression Assignment

1. Choose all valid identifiers
   1. int int
   2. int \_numvalue
   3. float price\_money
   4. char name1234567890123456789012345678901234567890
   5. char name value
   6. char $name
2. a. invalid, because keywords cannot be used as variables.

b. valid

c. valid

d. valid, because there is no rule on how long a identifier can be. However, you may run into problems in some compilers if the variable name is longer than 31 characters.

e. Invalid, because variable cant contain space.

f. Invalid, need not start with $.

1. What is the meaning of the following keywords, show the usage
   1. auto
   2. extern
   3. volatile
   4. sizeof
   5. const
2. a. **auto**: Declares a variable with automatic storage duration.

b. **extern**: Declares a variable or function defined in another file or scope.

c. **volatile**: Tells the compiler that a variable's value may change outside the program’s control.

d. **sizeof**: Returns the size in bytes of a variable or data type.

e. **const**: Declares a variable whose value cannot be modified after initialization.

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

Can you manipulate the contents of ptr? Why?

A. No, because ptr points to a string literal, which is stored in read-only memory, so modifying it is not allowed.

Can you manipulate the contents of arr? Why?

A. Yes, because arr is a writable array initialized with a copy of the string literal, so its contents can be modified.

Which one of the above is a string literal?

A. it is the string literal in both cases. It is stored in a special section of memory, but how it is used (via a pointer or an array) determines whether it is modifiable or not.

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);

}

A. 5 4

4 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;

}

1. The compile time error is due the use of const keyword while declaring num, if we remove that the will works without error.

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);

}

1. f1= 3.00

f2= 3.33

f3 =3.00