Lab 8 Pointer (2)

In this lab, the length of all the input cstrings are less than 100.

Question-1.
Write a program including a function called **deleteStr** (char* str, char delete_char). The return type is char pointer. The function is used to delete all the given delete_char existing in the given input string str. Use dynamic allocation to arrange space for the return cstring.

Note:
1. Use cin.get (delete_char) to get the input character because the char could be space.
2. You may first traverse the string str to record the number of given chars, making it convenient to set the size of dynamic array.
3. You can assume that the input strings str and char delete_char are not null.
4. Spaces can exist at the beginning or end of the input string.
5. No need to output the comment "//The input char is space" showed in the example 1.

Expected Outcomes:

Example 1

```
Enter the input string:
we are happy
Enter the input delete_char:
_       // The input char is space
The modified string is: wearehappy
```

Example 2

```
Enter the input string:
what date is it today
Enter the input delete_char:
t
The modified string is: wha dae is i oday
```

Example 3

```
Enter the input string:
count the number of it
Enter the input delete_char:
u
The modified string is: cont the nmber of it
```

Question-2.

[Will be marked. Your solution needs to pass ALL the E-QUIZ test cases of this question to get the mark for this lab; Otherwise, there is no mark for this lab.]

Write a program including a class called Candy. This 'Candy' class contains three members, 'price' of type int, 'amount' of type int, and 'name' of type char*. The default values of these three are 10, 20, and "strawberry", respectively. The design of class (named Candy) should include:

a) Three public members price, amount and name, which present information.
b) A default constructor (i.e. Candy()) to initialize the information of Candy.
c) A parameter constructor (i.e. Candy(int, int, char*)) to initialize the information for corresponding Candy.
d) A copy constructor (i.e. Candy(const Candy &other)) to initialize the information of one Candy with the information of another Candy.
e) A destructor (i.e. ~Candy()) to release the memory allocated to name.

Hint. Write code in the main() function for implementation.

Note: These three test cases will be generated by user input.

Expected Outputs

Example 1

`Which constructor to use (1: default, 2: parameterized, 3: copy)?`
`1`
`c1: name-strawberry; price-10; amount-20`

Example 2

`Which constructor to use (1: default, 2: parameterized, 3: copy)?`
`2`
`Enter name:`
`Green_Tea`
`Enter price:`
`12`
`Enter amount:`
`23`
`c1: name-Green_Tea; price-12; amount-23`

Example 3

`Which constructor to use (1: default, 2: parameterized, 3: copy)?`
`3`
`Enter name:`
`Green_Tea`
`Enter price:`
`23`
`Enter amount:`
`34`
`c1: name-Green_Tea; price-23; amount-34`
`c2: name-Green_Tea; price-23; amount-34`