Credit Card Processing

One of the minor annoyances of online shopping is that many web sites require you to enter a credit card without spaces or dashes, which makes double-checking the number rather tedious. How hard can it be to remove dashes or spaces from a string? Not hard at all, as this worked example shows.

Credit Card **Information** *(all fields are required)*	
We Accept:	MasterCard VISA AMERICAN EXPRESS Cards
Credit Card Type:	⌄
Credit Card Number:	
	(Do not enter spaces or dashes.)

Step 1   Decide what work must be done *inside* the loop.

In the loop, we visit each character in turn. You can get the ith character as

```
string ch = credit_card_number.substr(i, 1);
```

If it is not a dash or space, we move on to the next character. If it is a dash or space, we remove the offending character.

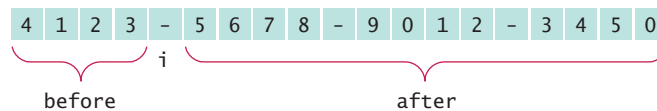> **Loop**
>   ch = the ith character of credit_card_number
>   **If** ch is a space or dash
>   Remove the character from credit_card_number.
>   **Else**
>   Increment i.

You may wonder how to remove a character from a string in C++. Here is the procedure for removing the character at position i: Take the substrings that end before i and start after i, and concatenate them.

4	1	2	3	–	5	6	7	8	–	9	0	1	2	–	3	4	5	0

before   i   after

```
string before = credit_card_number.substr(0, i);
string after = credit_card_number.substr(i + 1);
credit_card_number = before + after;
```

Note that we do *not* increment i after removing a character. For example, in the figure above, i was 4, and we removed the dash at position 4. The next time we enter the loop, we want to reexamine position 4 which now contains the character 5.

Step 2   Specify the loop condition.

We stay in the loop while the index i is a valid position. That is,

```
i < credit_card_number.length()
```

Step 3   Determine the loop type.

We don't know at the outset how often the loop is repeated. It depends on the number of dashes and spaces that we find. Therefore, we will choose a while loop. Why not a do loop? If we are given an empty string (because the user has not provided any credit card number at all), we do not want to enter the loop at all.

Step 4 Process the result after the loop has finished.

In this case, the result is simply the string.

Step 5 Trace the loop with typical examples.

The complete loop is

```
i = 0
While i < credit_card_number.length()
    ch = the ith character of credit_card_number.
    If ch is a space or dash
        Remove the character from credit_card_number.
    Else
        Increment i.
```

It is a bit tedious to trace a string with 20 characters, so we will use a shorter example:

credit_card_number	i	ch
4-56-7	0	4
4-56-7	1	-
456-7	1	5
456-7	2	6
456-7	3	-
4567	3	7

Step 6 Implement the loop in C++.

Here's the complete program, ch04/ccnumber.cpp.

```cpp
#include <iostream>
#include <string>

using namespace std;

int main()
{
   string credit_card_number = "4123-5678-9012-3450";

   int i = 0;
   while (i < credit_card_number.length())
   {
      string ch = credit_card_number.substr(i, 1);
      if (ch == " " || ch == "-")
      {
         string before = credit_card_number.substr(0, i);
         string after = credit_card_number.substr(i + 1);
         credit_card_number = before + after;
      }
      else
      {
         i++;
      }
   }

   cout << credit_card_number << endl;

   return 0;
}
```