Nested Loops and Common Loop Algorithms

Today

- Common loop algorithms for strings
- Nested Loops

Due this week

- **Homework 3**
  - Write solutions in VSCode and paste in Autograder, **Homework 3 CodeRunner**.
  - Zip your .cpp files and submit on canvas **Homework 3**.
  - Review file nameand header requirements
- Start going through the textbook readings and watch the videos
  - Take **Quiz 4**.
- Participation: 3-2-1
- Check the due date! **No late submissions!!**

Common Loop Algorithms

Traversing a string with loops

```
int main()
{
    string str = "ABC";
    for(int i=0; i < str.length();i++)
    {
        cout << str[i] << endl;
    }
    return 0;
}
```

Traversing a string with loops

```
int main()
{
    string str = "ABC";
    for(int i=0; i < str.length();i++)
    {
        cout << str[i] << endl;
    }
    return 0;
}
```

Common Loop Algorithms: Counting Matches

```
//Counting chars in a string
int spaces = 0;
for (int i = 0; i < str.length(); i++)
{
      if (str.substr(i, 1)== " ")
      {
            spaces++;
      }
}
```

```
//Counting words in a user input sequence
int short_words = 0;
string input;
while (cin >> input)
{
      if (input.length() <= 3)
      {
            short_words++;
      }
}
```

Common Loop Algorithms: Finding First Location

```
//Find the location in a string of first space char

bool found = false; //flag=false says "not found yet"
int position = 0;

while (!found && position < str.length())
{
    string ch = str.substr(position, 1);
    if (ch == " ")
    {
            found = true;
    }
    else
    {
            position++;
    }
}
```

Common Loop Algorithms: Prompting Until Matched

```
//Repeat prompt until user enters valid value

bool valid = false; //input not valid yet
double input;  //declare input var outside loop,
               //so it will persist after loop exit
while (!valid)
{
   cout << "Please enter a positive value < 100: ";
   cin >> input;
   if (0 < input && input < 100)
         { valid = true; }
   else
         { cout << "Invalid input." << endl; }
}
```

Common Loop Algorithms: Min and Max

```
//Save the min and max values of user input list
// This is a merger of the min and max loops from book

double largest, smallest;
double input;
cin >> largest; //get first value to use in loop
smallest = largest; // copy it.
// If only 1 entry, it is both smallest and the largest

while (cin >> input)
{
   if (input > largest)
   { largest = input; }
   else if (input < smallest)
   { smallest = input; }
}
```

Common Loop Algorithms: Comparing Adjacent Values

```
//Find adjacent duplicates of user input list
// In a later chapter, we'll show how to use arrays to
// find non-adjacent duplicates

double input;
double previous; //to keep track of prior entry
cin >> previous; //first entry becomes first previous
while (cin >> input)
{
   if (input == previous)
   {
         cout << "Duplicate input" << endl;
   }
   previous = input; //save it to compare to next input
}
```

Worked Example 4.1: Loop to Remove Chars from `string`

```
// worked_example_1/ccnumber.cpp
// Removes all spaces or dashes from a string

Two options:

1.Create a new string that will have the answer
   • add/concatenate in there only the characters you
     want to keep

2.Modify the original string variable
   • Keep reassigning new values to the original string
     by piecing together substrings from before and after
     any character you don't want
```

Worked Example 4.1: Loop to Remove Chars from `string`

```
#include <iostream>                    // worked_example_1/ccnumber.cpp
#include <string>                      // Removes all spaces or dashes from a 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 == "-") //must remove char
        {
            string before = credit_card_number.substr(0, i);
            string after = credit_card_number.substr(i + 1);
    credit_card_number = before + after;
        }
        else // no need to remove it, go to next char
        { i++; }
    }
    cout << credit_card_number << endl;
    return 0;
}
```