Class 25

string methods + Files + Bubble Sort
Class Methods

• Class type has methods, which are special functions
• Call method:
  • VarName.MethodName(Arguments)
substr() – from specified position

• Extracts a substring starting at a specified position
• Given a string and a position
• substr(position)

string example = “Queens College”;
string substr1 = example.substr(7);
cout << substr1; // prints College
substr() – specific length, from specified position

- Extracts a substring of a specified length (number of characters) from a string, starting at a specified position
- Given a string, a position and a length of the substring
- `substr(position, length)

```cpp
string example = "Queens College";
string substr1 = example.substr(0, 6);
cout << substr1; // prints Queens
```
Insert

- `insert(position, newText)`

- **Example:**

  ```java
  string s = "College";
  s.insert(0, "Queens "); // s now stores Queens College
  ```
Erase, Replace

- `erase(position, amount)`
- `replace(position, amount, newText)`
- Example:

    string s = “Queens College”;
    s.erase(6, 2); // s now stores Queensollege
    s.replace(0, 6, “C”); !! s now stores College
Append

- append(addition)
- Example:

```java
string s = "Queens";
s.append(" College"); // s now stores Queens College
```
Input from/Output to files

- This is performed using streams
- A stream on a computer performs input/output operations
- It can be viewed as either a destination or a source of indefinitely long characters
- C++ comes with a library called fstream that includes methods for dealing with files
Input from/Output to files

• Class types:
  • ifstream – used to read information from files
  • ofstream – used to create files and to write information to files

• #include<fstream>

• Class methods:
  • .open(fileName) – connects a variable to a file
  • .is_open() – checks to see if file is open
  • .close() – closes a file

• ifstream also has .eof() and .get()
Example

```cpp
ofstream f;       // f is a variable to access our output file
f.open(“out.txt”);   // connects f to file out.txt
f << “Hello” << endl;   // puts output into the file
f.close();          // to properly close the file
```
Example 1

```cpp
#include<iostream>
#include<fstream>
using namespace std;

int main(){
    ofstream f;
    f.open("out.txt");
    if (!f.is_open()) {
        cout << "Cannot open file. See ya." << endl;
        return 0;
    }
    f << "Hello" << endl;  // insertion operator
    f << "World" << endl;
    f.close();
    return 0;
}
```
ifstream

• Considerations when reading files:
  • Have you reached the end of the file (is there no more data left to read)?
    • Answer this question with .eof()

• What if you would like to read the input character by character?
  • Use .get() to obtain the next character in the file
  • This also reads whitespaces, such as spaces, new lines, etc.
  • Used to obtain very detailed input
Example

```cpp
ifstream f;
f.open("out.txt");
string s;
f >> s;     // Extracts first string in file connected to f
f.close();
```
Example 2

```cpp
#include<iostream>
#include<fstream>
using namespace std;

int main(){
    ifstream f;
    f.open("animals.txt");
    string x, y;
    f >> x; // extraction operator
    cout << "The first string in your file is " << x << endl;
    f >> y;
    cout << "The next string in your file is " << y << endl;
    f.close();
    return 0;
}
```
Example

```cpp
ifstream f;
f.open("animals.txt");
char x = f.get(); // get next character
while(!f.eof()){  // while you have not yet reached the end of the file
    cout << x;   // print character to monitor
    x = f.get(); // get next character
}
f.close();
```

The above goes through file f character by character and prints whatever it sees to the monitor
Example 3

```cpp
#include<iostream>
#include<fstream>
using namespace std;

int main()
{
    ifstream f;
    f.open("animals.txt");
    char x = f.get();  // get next character
    while(!f.eof()){   // while you have not yet reached the end of the file
        cout << x;     // print character to monitor
        x = f.get();   // get next character
    }
    f.close();
    return 0;
}
```
Bubble Sort

Repeatedly swap adjacent elements in an array if they are not in the right order.

```c
void bubbleSort(int a[], int size){
    for (int i = 0; i < size - 1; i++)
        for (int j = 0; j < size - 1 - i; j++)
            //Sort in descending order
            if (a[j] < a[j+1]) swap(a[j], a[j+1]);
}
```