

Lab Tutorial 4

String Processing in C++

- Class **String** contains several useful functions to perform string processing in C++.

```
#include <string>
```

```
#include <iostream>
```

```
using namespace std;
```

```
int main(){
```

```
    string s1;
```

```
    s1 = "This is a string";
```

```
    cout << s1 << endl;
```

```
    return 0; }
```

String Processing in C++

Class string takes care of several important operations performed on strings. For example:

- compare two strings
- search for a specific character in a string
- swap the values of two strings
- concatenate two strings
-

Comparing two Strings

```
string s1 = "first string";  
string s2 = "string";  
cout << s1.compare(s2) << endl;
```

- 0: if equal.
- -1: Not equal. 1st non matching character in Var is less in value based on ASCII table than in compare string.
- +1: Not equal. 1st non matching character is greater in value based on ASCII table.

Getting the String Size

```
string s1 = " string";  
cout << s1.size() << endl;
```

- Returns the size of memory assigned to the string
- is equivalent to length() and capacity() functions.

Read a Character from String

```
string s1 = " string";  
cout << s1.at(1) << endl; // output: 't'  
cout << s1.at(4) << endl; // output: 'n'
```

- Returns the character at a specific location of the string.
- Is equivalent to `s[i]`, if `s` is defined by `char[]`.

Search for a String

```
string s1 = "this is a string";  
string s2 = "is";  
cout << s1.find(s2) << endl; // output: 2  
cout << s1.find("is a") << endl; // output: 5
```

- Returns the *first* occurrence index of the given string.
- Returns **string::npos** if not found any occurrences.

Get a substring

```
string s1 = "this is a string";  
cout << s1.substr(5, 8) << endl; // output: "is a str"
```

s1.substr(5)

- substr(pos, len) returns the substring starting from **index pos**, with the **length len**, from the string.
- And, the **len** will be the size of that new string.

Swap Two Strings

```
string s1 = "this is a string";  
string s2 = "this is another string";  
s1.swap(s2)  
cout << s1 << endl; // output: "this is another string"  
cout << s2 << endl; // output: "this is a string"
```

- Swaps the values of the two strings.

Quick Hint!

- Given a character which identifies a number between 0 and 9.
- What is the easiest way to convert it to an integer value?

- Example:

```
char a = '5';
```

```
int b;
```

```
// desired output: b = 5
```

```
b = a - '0';
```

Exercise 1

- Use this hint to convert the integer given in a string, to its actual value.
- Example:

Input: `string num_str = "123456";`

Output: `int number = 123456;`

Exercise 2

- Given a sentence (string whose words are separated by space), write a code to separate out all the words.
- Example:

Input: string sentence = “this is a sample sentence”;

Outputs: “this”, “is”, “a”, “sample”, “sentence”

Stack

- Stack is one of the most popular data structures used in several application.
- **LIFO: Last In, First Out**
- Each element which is most recently inserted (**pushed**), is the one who is first removed (**popped**).
- Used by compilers to compile the code, by web browsers to handle “back” and “forward”, and by text editors to handle “undo” and “redo”
- Example:
 - `push(1), push(2), push(3), push(4)`
 - `pop()` → outputs 4
 - `push(5)`
 - `pop()` → outputs 5

Queue

- Queue is another popular data structure which works in an opposite manner:
- **FIFO: First In, First Out**
- Each element which is first inserted (**enqueued**), is the one who is first removed (**dequeued**).
- Most intuitive example is a group of people standing in a line!
- Example:
 - enqueue(1), enqueue(2), enqueue(3), enqueue(4)
 - dequeue() → outputs 1
 - enqueue(5)
 - dequeue() → outputs 2

Exercise 3

- Implement a simple stack of integers using a standard array.
- You should develop the following two functions:
 1. `push(int newElement)`
 2. `pop()`

Any Questions?