Mittuniversitetet
MID SWEDEN UNIVERSITY

Lecture #3
Some predefined Functions, Characters, Strings

Chapter: 3.2, 3.2.3, 4.11.3

Objectives

Mittuniversitetet
MID SWEDEN UNIVERSITY

- To be able to use predefined functions.
- To represent and use characters using the **char** type.
- To represent special characters using the escape sequences.
- To cast a numeric value to a character and cast a character to an integer.
- To compare and test characters.
- To test and convert characters using the C++ character functions.
- To represent strings using the **string** type and introduce objects and instance functions.
- To use the subscript operator for accessing and modifying characters in a string.
- To use the + operator to concatenate strings.
- To compare strings using the relational operators.
- To read strings from the keyboard.

Some useful predefined functions

Mittuniversitetet
MID SWEDEN UNIVERSITY

ceil(x) x is rounded up to its nearest integer. This integer is returned as a double value.
floor(x) x is rounded down to its nearest integer. This integer is returned as a double value.

max(2, 3)	returns 3
max(2.5, 3.1)	returns 3.1
min(2.5, 3.1)	returns 2.5
abs(-2)	returns 2
abs(-2.5)	returns 2.5

PreDefined.cpp

Character Data Type

Mittuniversitetet
MID SWEDEN UNIVERSITY

char letter = 'A'; (ASCII)

char numChar = '4'; (ASCII)

NOTE: The increment and decrement operators can also be used on char variables to get the next or preceding character. For example, the following statements display character b.

```
char ch = 'a';
std::cout << ++ch;
```

Reading Characters and casting to integer

Mittuniversitetet
MID SWEDEN UNIVERSITY

Write a program that can read characters, print them out as char and integer.

```
int i = '2' + '3'; // '2' has ascii code 50 and '3' has 51
std::cout << "i is " << i << std::endl; // i is integer value of 101

int j = 2 + 'a'; // 'a' ascii value 97
std::cout << "j is " << j  << std::endl;

std::cout << j << " is the ASCII code for character " << static_cast<char>(j);
```

CastingCharacters

Character Functions

Mittuniversitetet
MID SWEDEN UNIVERSITY

Function	Description
`isdigit(ch)`	Returns true if the specified character is a digit.
`isalpha(ch)`	Returns true if the specified character is a letter.
`isalnum(ch)`	Returns true if the specified character is a letter or digit.
`islower(ch)`	Returns true if the specified character is a lowercase letter.
`isupper(ch)`	Returns true if the specified character is an uppercase letter.
`isspace(ch)`	Returns true if the specified character is a whitespace character.
`tolower(ch)`	Returns the lowercase of the specified character.
`toupper(ch)`	Returns the uppercase of the specified character.

Converting a Lowercase to Uppercase

Mittuniversitetet
MID SWEDEN UNIVERSITY

Write a program that prompts the user to enter a lowercase letter and converts into corresponding uppercase letter.

```
std:: cin >> lowercaseLetter;

std:: cout << "The corresponding uppercase letter is "
      << std::toupper(lowercaseLetter) << std:: endl;
```

ToUppercase

Case Study: Generating Random Characters

Mittuniversitetet
MID SWEDEN UNIVERSITY

Computer programs process numerical data and characters. You have seen many examples that involve numerical data. It is also important to understand characters and how to process them.

Every English character has a unique ASCII code between 0 and 127.

To generate a random character is to generate a random integer between 0 and 127. You can use the **srand(seed)** function to set a seed and use **rand()** to return a random integer. You can also use it to write a simple expression to generate random numbers in any range.

Generating Random Characters (cont.)

Mittuniversitetet
MID SWEDEN UNIVERSITY

```
rand() % 10
```
→ Returns a random integer between 0 and 9.

```
50 + rand() % 50
```
→ Returns a random integer between 50 and 99.

Generally:

```
a + rand() % b
```
→ Returns a random number between a and a + b, excluding a + b.

The string Type

Mittuniversitetet
MID SWEDEN UNIVERSITY

std::string s;

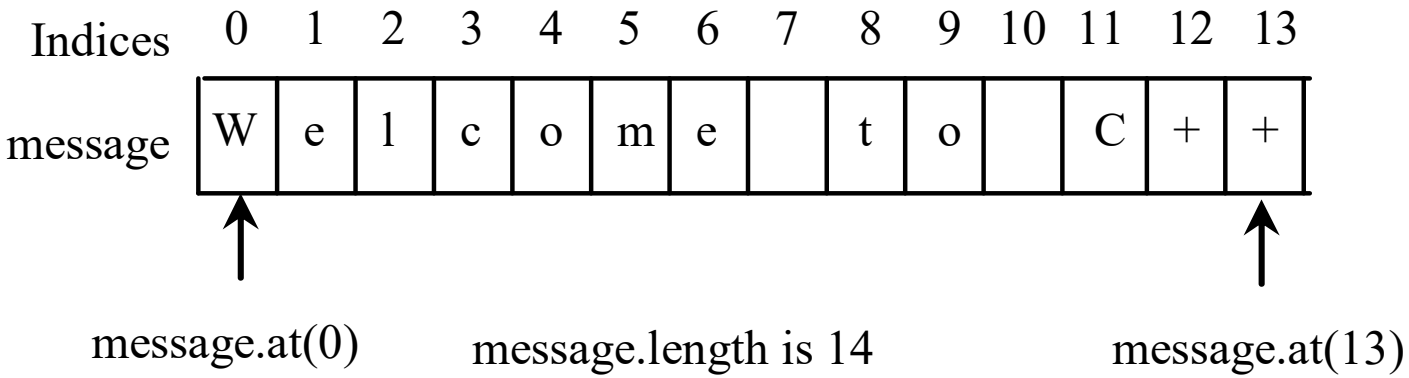
std:: string message = **"Programming is fun"**; //Less good

std:: string message1 (**"Programming is fun"**); // better than message above

std:: string message2 {**"Programming is fun"**}; //Not exactly the same as the two above!, But this is to prefer! C++11

Function	Description
`length()`	Returns the number of characters in this string.
`size()`	Same as length();
`at(index)`	Returns the character at the specified index from this string.

String Subscript Operator



For convenience, C++ provides the subscript operator for accessing the character at a specified index in a string using the syntax **stringName[index]**. You can use this syntax to retrieve and modify the character in a string.

String Subscript Operator

Mittuniversitetet
MID SWEDEN UNIVERSITY

```
string s { "ABCD“};
s[0] = 'P’;
std::cout << s[0] << std::endl;
```

Concatenating Strings

Mittuniversitetet
MID SWEDEN UNIVERSITY

```
std::string s3 = s1 + s2;

message += " and programming is fun";

message.append(" always ");
```

Comparing Strings

Mittuniversitetet
MID SWEDEN UNIVERSITY

You can use the relational operators **==**, **!=**, **<**, **<=**, **>**, **>=** to compare two strings.

This is done by comparing their corresponding characters on by one from left to right.

For example:

```
std::string s1 { "ABC"};

std::string s2  {"ABE"};

std::cout << (s1 == s2) << std::endl; // Displays 0 (means false)

std::cout << (s1 != s2) << std::endl; // Displays 1 (means true)

std::cout << (s1 > s2) << std::endl; // Displays 0 (means false)

std::cout << (s1 >= s2) << std::endl; // Displays 0 (means false)

std::cout << (s1 < s2) << std::endl; // Displays 1 (means true)

std::cout << (s1 <= s2) << std::endl; // Displays 1 (means true)
```

Reading Strings

Mittuniversitetet
MID SWEDEN UNIVERSITY

Version 1:

```
std::string city;
std::cout << "Enter a city: ";
std::cin >> city; // Read to string city
std::cout << "You entered " << city << std::endl;
```

Reading_Strings

Version 2:

```
std:: string city;
std:: cout << "Enter a city: ";
std:: getline(std::cin, city, '\n'); // Same as getline(std::cin, city)
std:: cout << "You entered " << city << std:: endl;
```

Example: why didn't I get a chance to type?

Mittuniversitetet
MID SWEDEN UNIVERSITY

Write a program that prompts the user to enter firstname, lastname and a lucky number and displays them.

MixedInputs

Input and Output Redirections

Mittuniversitetet
MID SWEDEN UNIVERSITY

if you have a large number of data to enter, it would be cumbersome to type from the keyboard. You may store the data separated by whitespaces in a text file, say **input.txt**, and run the program using the following command:

In Linux:

./ReadValues < input.txt

In Windows:

ReadValues < input.txt

Thank you