Name: Kormoua Khongmeng
Neptun Code: I3MLPQ

Basic of Programming 2

Laboratory report 10

Task

b) At first (line 10), a string “str1” is created and assigned by an array of characters. In this case a conversion constructor which convert an array of characters to a String is called. this is an implicit type constructor call, done by the compiler.

After that (line 11), a printf() function is called and received a String as an argument, at this point the conversion operator is called, which a string class “str1” is converted back into an array characters again before the printf() function print an array of character into the screen.

After printing the program reaches to a point where a conversion constructor is called again(line 12), just like before. It converts the array of characters into a String.

After that when the program reach to where we will append a new array of character to the string “str1” (line 13), the conversion constructor which converts an array of characters into a string is called again and convert an array of characters “James Bond.” Into a string first then after that pass this a string “James Bond” into a operator+= overloading next. To append a previous string “My name is Bond. ” with the new string “James Bond.”.

Then when the program are done with appending 2 string, a printf() function is called again (line 14). At this point, a conversion operator is called again to convert a string into an array of characters and return this array of characters to printf() function.

Then, when the program arrive to if comparation if statement (line 16), a conversion constructor is called again to convert those 2 array of characters to strings, then an operator== overloading and operator!= overloading are called next to compare string “str1” with those 2 strings that are just converted from an array of characters, but this time, a conversions are called explicitly. Then the program goes to the end and terminate itself.

c) If we leave out the explicit type conversion from the mentioned piece of code will generates a compiler error. The cause of error is an ambiguous when using overloaded operator ‘==’, because the operands in the mentioned code is ‘String’ and ‘const char*’ but there is no operator== overloading to handle this case precisely. We have created operator== overloading to compare ‘String’ and ‘String’. But there is a built-in operator== which compare ‘const char*’ and ‘const char*’ as well. So when the operands are ‘String’ and ‘const char*’ the compiler doesn’t know which operator== to used, should the ‘const char*’ be converted to ‘String’ or should the ‘String’ be converted to ‘const char*’. This causes an ambiguous and causes an error.