

Programming Design, Spring 2016

Homework 12 Solution

Solution provider: Wei-Hung Liao
Department of Information Management
National Taiwan University

Problem 1

- (a) The member function `find` finds the first occurrence of the specified c++ string, c string, or character after the given position of the original string and return the position if it finds it, `string::npos` otherwise. If the type of the input is c string (character array), we can also set the length of the c string we want to match. Though the parameters are all the same as the function `find`, function `find_first_of` is totally different from `find`. The function `find_first_of` find the first occurrence of **any character** in the given string. The function `find_first_not_of` do the opposite thing of `find_first_of`. It find the first character that does not exist in the given string.

In short, the difference between them is they find the first occurrence of the given string, the character in the given string, and the character no in the given string.

(b)

```
bool string::operator==(const string& s) const {
    if(this->compare(s)==0)
        return true;
    return false;
}

bool string::operator!=(const string& s) const {
    if(this->compare(s)!=0)
        return true;
    return false;
}

bool string::operator<(const string& s) const {
    if(this->compare(s)<0)
        return true;
    return false;
}

bool string::operator<=(const string& s) const {
    if(this->compare(s)<=0)
        return true;
    return false;
}

bool string::operator>(const string& s) const {
    if(this->compare(s)>0)
        return true;
    return false;
}

bool string::operator>=(const string& s) const {
    if(this->compare(s)>=0)
        return true;
    return false;
}
```

(c)

```
string& clear(size_t pos = 0, size_t len = npos) {
    return this->erase(0, this->length);
}
```

(d)

```
size_t string::getSpaceCount ( const size_t from , const size_t to ) const {
    int count=0;
```

```
        for(int i=from; i<=to; i++) {  
            if(this->c_str()[i]==' ') {  
                count++;  
            }  
        }  
        return count;  
    }  
}
```

The first two `const` indicate that this function won't modify the two parameters. Since the two parameters are called by value, the two `const` are useless (even without the two `const`, the two variable outside the function won't be modified). The last `const` tells the compiler that this member function won't modify the value of any member variable. The compiler then allows constant object use this function. The reason that there is no `const` at the beginning of the function is that we may want to assign the return value to non-constant variables.

Problem 2

Please see the attached CPP file.

Problem 3

Please see the attached CPP file.