# Hands-on Experiment # 6 : Worksheet

Section\_\_\_\_\_2\_\_\_\_\_\_\_\_\_ Date\_\_\_\_\_\_\_\_\_\_\_\_\_22/02/2018\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

No more than 3 students per one submission of this worksheet.

Student ID \_\_\_\_\_\_\_\_\_6031851521\_\_\_\_\_\_\_\_ Name\_\_\_Sarun Nuntaviriyakul\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Student ID \_\_\_\_\_\_\_\_\_6031848721\_\_\_\_\_\_\_\_ Name\_\_\_Watcharin Kriengwatana\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Student ID \_\_\_\_\_\_\_\_\_6031847021\_\_\_\_\_\_\_\_ Name\_\_\_Wasuthon Klyhirun\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

## Part A: Complex Conditions

An online service lets its users register for a new account by themselves. Each user can choose a username as well as a password. Here are the rules on picking a username and setting the associated password.

Username

* A username must be at least 4 characters in its length but must not exceed 255 characters.
* The first character of a username must be English alphabets. It can be either a lowercase letter (a-z) or an uppercase letter (A-Z).
* No spaces are allowed in a username.
* No backslashes are allowed in a username.
* Dots (.) can be used but no dots can be adjacent to one another.

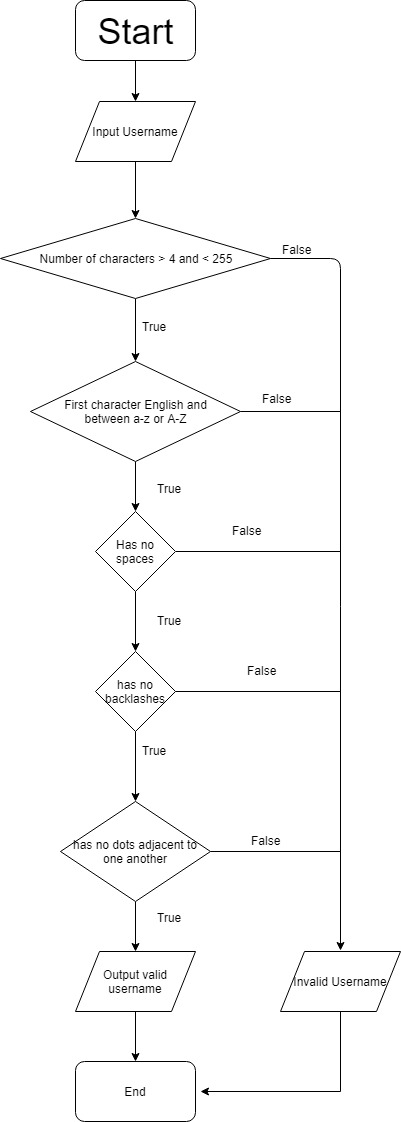
Password

* A password must be at least 4 characters in its length but must not exceed 255 characters.
* A password must not contain the username regardless of their cases. This means that when checking whether the username is in the password, lowercase and uppercase letters are considered the same if they are the same alphabets. (E.g. A password is not allowed to contained “JaSoN” if the username is “jason”)
* If the length of a password is less than 8 characters, the password must contain at least 2 digits (0-9) and at least one of the following symbols: ! @ # $ %.
* An asterisk (\*) cannot be used in a password.
* No spaces are allowed in a password.
* No backslashes are allowed in a password.

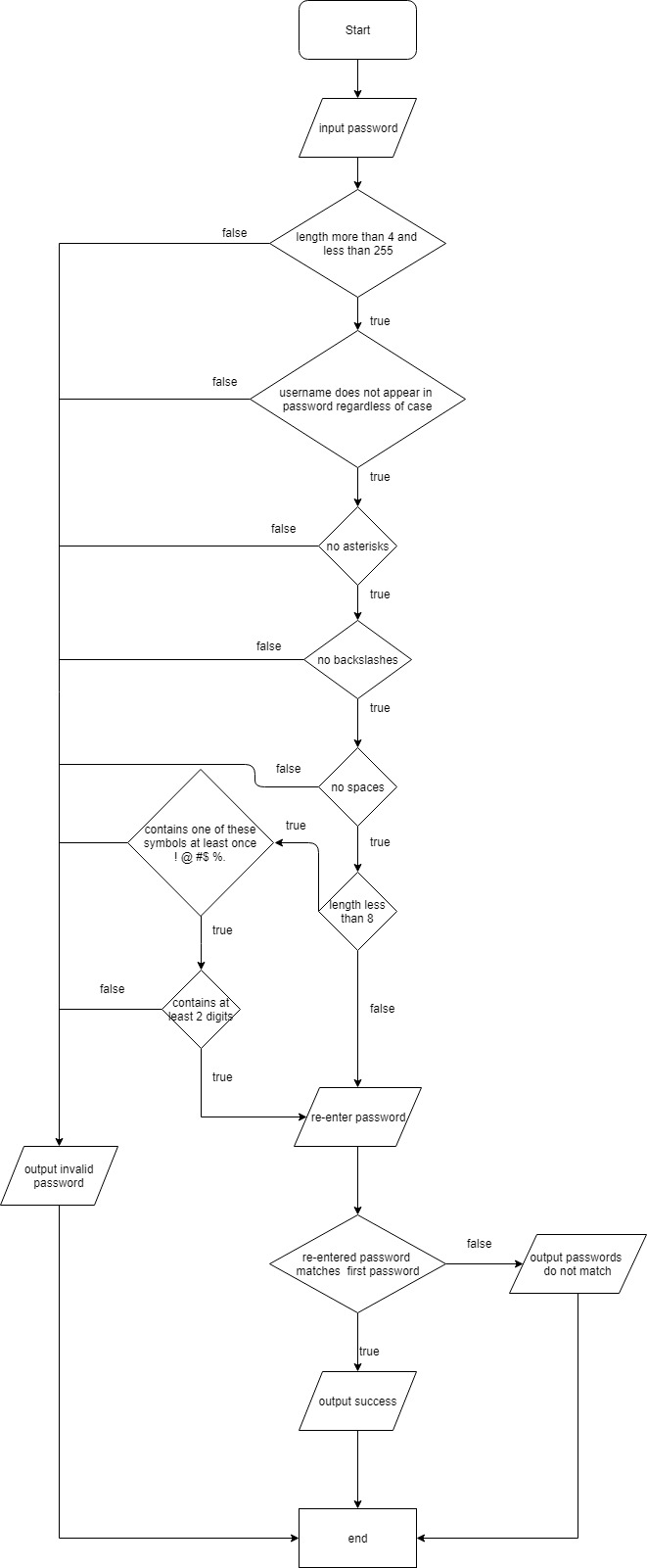
Determine whether the username/password combination in each row of the following table conforms to the rules above. If not, state the reason why.

|  |  |  |  |
| --- | --- | --- | --- |
| Username | Password | Conform to rules (Y/N) | If not, why? |
| Broccoli\_Duke | 5%x2 | No | Space in ID |
| Anna | itisANNA | No | Anna in password |
| Anna.Stone.2 | BGiskj129Dk8960 | Yes |  |
| Anna……Stone | qwertyUIDFO0001\_3 | No | Has adjacent dots |
| Anna Stone | ABCDEFGHIJK | No | Space in id |
| 5640021321 | 1234567890 | No | First ID is a digit |
| King.of.StarCraft | zerg | No | No 2 digit and Symbol |
| King.of.StarCraft2 | \*\*\*\*\*\*\*\* | No | Asterisk cannot be use |
| King.of.StarCraft3 | 09638 | No | No symbol when less than 8 |
| Boxster999 | 0000 0000 | No | There are space in password |
| Crafty\Random | Complicated\_password | No | There are backslash |

Draw a flowchart showing the process to be performed for verifying whether *an input username* conforms to the rules.



Draw a flowchart showing the process to be performed for verifying whether *an input username/password combination* conforms to the rules. Assume that the username has already been set so that it conforms to the rules.



Write a Java program that performs the following steps.

1. Ask the user to set a username.
   1. If the username conforms to the rules, proceed to password setting.
   2. If the username does not conform to the rules, the program ends.
2. Ask the user to set a password.
   1. If the password conforms to the rules, proceed to password confirmation.
   2. If the password does not conform to the rules, the program ends.
3. Ask the user to re-enter the password for confirmation.
   1. If the password is the same as the one in 2, show a message saying that the account has successfully been created.
   2. If the password is different from the one in 2, the program ends.

Methods in the String class should be useful in this program. Check its API specification at <http://docs.oracle.com/javase/7/docs/api/java/lang/String.html>

List your source code below.

import java.util.Scanner;

public class idPassChecker {

public static boolean twoNum(String id) {

int len = id.length();

int count = 0;

for(int i = 0 ; i < len ; i++){

char c = id.charAt(i);

if(Character.isDigit(c)) count ++;

}

return (count>=2 ? true:false);

}

public static void main(String [] args) {

Scanner sc = new Scanner(System.in);

String username;

String password;

boolean badUsername = true;

System.out.println("Please create your username :");

username = sc.nextLine();

boolean backSlash = username.contains("\\");

int len = username.length();

boolean dbleDot = username.contains("..");

boolean space = username.contains(" ");

if(len >= 4 && len <= 255) {

if( (username.charAt(0) >= 'a' && username.charAt(0) <= 'z') || (username.charAt(0) >= 'A' && username.charAt(0) <= 'Z') ) {

if(!space){

if(!backSlash){

if(!dbleDot){

System.out.println("Good.");

badUsername = false;

}

}

}

}

}

if (badUsername){

System.out.println("Invalid username");

System.exit(0);

}

System.out.println("Please create your password");

password = sc.nextLine();

int passlen = password.length();

boolean atUser = password.toLowerCase().contains(username.toLowerCase());

boolean asterisk = password.contains("\*");

boolean backSlash2 = password.contains("\\");

boolean space2 = password.contains(" ");

boolean badPassword = true;

if (passlen >= 4 && passlen <= 255){

if(!atUser){

if(!asterisk){

if(!backSlash2){

if(!space2){

if (passlen <= 8){

if(password.contains("!")||password.contains("@")||password.contains("#")||password.contains("$")||password.contains("%")) {

if(twoNum(password)){

badPassword = false;

System.out.println("Good");

}

}

}

else {

badPassword = false;

System.out.println("Good");

}

}

}

}

}

}

if(badPassword) {

System.out.println("Invalid password");

System.exit(0);

}

System.out.println("Please re-enter your password.");

String password2 = sc.nextLine();

if (!password2.equals(password)) {

System.out.println("Wrong");

System.exit(0);

}

System.out.println("Finish!");

}

}

Test the program with the username/password combinations listed in the first table.

Does your program verify those combinations correctly?

Yes.

Submit this worksheet (by only one member of the group) via <http://www.myCourseVille.com> (Assignments > Hands-on Experiment # 6) before noon of the day after your lecture.