```
 /* HW3
The purpose of this homework is to make you understand: classes and objects,
    methods , if statements and Loops, Ternary operators , ArrayList and
    generate random characters using Random class.

The idea of this HW is as follows :
A password must meet special requirements, for instance , it has to be of
specific length, contains at least 2 capital letters , 2 lowercase letters,
2 symbols and 2 digits. A customer is hiring you to create a class that
can be utilized for several clients of the customer.

The first object created will be for the first level clients which requires
a moderate level password which contains ( at least 1 capital letters ,
1 lowercase letters,1 symbol and 1 digit).
The first level should be created using a default constructor.
The length of any password should be at least 8 characters for all levels.
Hint, should be static.

The second object created will be for the second level clients which requires
a harder level password which contains ( at least 2 capital letters ,
2 lowercase letters,2 symbol and 2 digits).
The second level should be created using an overloaded constructor.
the length of the second level password should be at least 8 as it
required for any levels.

The third object created will be for the third level clients which requires
a harder level password which contains ( at least 3 capital letters ,
3 lowercase letters,3 symbol and 3 digits).
The third level should be created using an overloaded constructor.
The length of the third level password should be at least 12.

The forth object is an extreme case , if an object created using extreme case
settings, for instance, negative number or more than 3 of (capital letters,
lowercase.....etc.) ,then it should be revert back to the default settings
using the default constructor.

*/



import java.util.ArrayList;
import java.util.Random;
import java.util.Scanner;

public class Password2{


/*(1 point) Create a private static default password variable and set
```

```
    it to Def@u1tPa$$w0rd which meets the standards of accepted password*/

//(1 point) Create a private static int length of the password and set to 8.

// Private settings that should be met for each password instance/objects
//(1 point) Create a private variable to store the number of symbols

//(1 point) Create a private variable to store the number of capital letters

//(1 point) Create a private variable to store the number of lower case letters

//(1 point) Create a private variable to store the number of digits

// (1 point)Create a private variable to store the password

/** (6 points) Create the default constructor, set the default required number
 * for capital and lowercase letters, symbols, digits to 1
 * set the password variable to the default password
*/
    Password2()
    {   // default values


    }

    /**( 6 points) Create an overloaded constructor ,that takes
     * number of ( symbols , capital letters, small letters, digits)
     * these will be considered settings for initializing an instance of
     * the password set the global private variables (also known as data fields)
     *  to the passed in arguments
     */
    Password2 (  int numSymbols, int numCap, int numSmall, int numDigits )
    {
       /*  ------------------------------------------------
        HW4  (4 points)
        Make sure to prevent users from supplying settings that are
        invalid settings ( for instance , negative values or 0) or
        extreme cases like requiring more than 3 of (capitals,lowercase,
        symbols,digits) in case , an extreme case supplied,
        print a message that the settings supplied is not acceptable and
        apply the default setting.
        xxxxxxxxxxxxxxxxxxxxxxxxxxxxxTYPOxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
        xxxxxxUse the default constructor to apply the default settingsxxxx
        xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
        Apply the default settings like it is done in the defualt construcotr.
        ------------------------------------------------    */


    }
```

```
    /**(3 points) Create a method that takes a string password and check if
     * it is equal to the length specified then return true, false if not */
    public boolean validLength(String pass)
    {
        /*----------------------------------------
        HW4 ( 4 points)
           Use the ternary operator
        if the addition of settings numbers is less than 8 , then minimum is 8
        for instance ( 1 capital, 1 small , 1 symbol, 1 number) = 4 which is less
        than the default length required, so minimum in this case 8.
        if the addition of settings numbers is higher than 8 , then minimum is the
        addition of numbers.
        ---------------------------------------- */
      return true;
    }

     /*-----------------------------------------------------------------
    HW4 (5 points)  There are four methods that are similar to each other.
    one way to improve in here is to create one method that does thier actions.
    take the first method that checks how many symbols and name it to checkRange
   the method looks like this:
    public boolean checkRange( String pass,int required, char start,char end)
    this will reduce the lines of codes significantly.
    ----------------------------------------------------------------- */
     /**(10 points) Create a method that takes a string password ,
      * the method checks if the password has at least the
      * required number of symbols and return true, false if not


            // Declare a counter variable

            // Declare a boolean to hold the answer

           // Loop through the length of the password

         // Once you counted the required number, set answer to true and break


         // Using the Ascii table, check each index if in the range [32 -47] or
[58-64]


                    // increment the count


            // return the answer

        */
```

```
    /**(3 points) Create a method that takes a string password ,
    * the method checks if the password has at least the
    * required number of digits and return true, false if not
    * the style of this method will be similar to the previous method
    * use the range in the Ascii table [48 -57] for digits
    */



    /**(3 points) Create a method that takes a string password ,
    * the method checks if the password has at least the
    * required number of capital letters and return true, false if not
    * the style of this method will be similar to the previous method
    * use the range in the Ascii table [65 -90] for capital letters
    */


    /**(3 points)Create a method that takes a string password ,
    * the method checks if the password has at least the
    * required number of lowercase letters and return true, false if not
    * the style of this method will be similar to the previous method
    * use the range in the Ascii table [97 -122] for lowercase letters
    */


    /**(2 points) create a getter method to return the password*/


    /** (10 points) Create a setter method to set the password*/
    public void setPassword(  )
    {
        // Declare a String to hold a password

        // Declare a scanner object to receive a password from the keyboard

        // Declare a boolean variable to be set whenever a password meets the
settings.

        // Loop until the user provides a correct password

                // Prompt the user to enter the password , specify the requirements

       /*-HW4 ( 5 points)--------------------------------------------------

        //Print out 4 suggested password based on the settings initialized by the
constructor

        // Scan the next line and store in the String holding the password
```

```
    Rather than having a chained if statements which will print one
    error message only, a need to see  all error messages is required ,
    for instance , if the password provided doesn't have
    2 capital letters or 2 lower case or  if the password length
    less than requested. So,  independent if statements will do the job and
    print out all messages.

    /* If the password provided is not equal to the length required,
       then print out an error message*/

    /*  If the password does have similar consecutive characters
    then print out a message*/


    /*  if the password doesn't have the required number of capital letters
    then print out a message*/


    /*  if the password doesn't have the required number of lowercase letters
    then print out a message*/


    /*  if the password doesn't have the required number of symbols
    then print out a message*/


    /*  If the password doesn't have the required number of digits
    then print out a message*/

    /* if , password provided is correct
        set the global variable password to the new qualified password
        set the flag to false, to stop iterations
    */
     /* ------------------------------------------------HW4*/
    }

  /*-------------------------------------------------
HW4 ( 10 points)
Build a new method that suggests few passwords based on
the settings required , you have to use Random class which let you
generate random numbers, which will be used to generate random characters
that are within specified range.
The method should take a specified range and required charcters, it should
return an array encapsulating the answers.
you can utilize an arrayList as the array of answers will be of different
sizes based on the settings required.
-------------------------------------------------*/

/*-------------------------------------------------
```

```
HW4 ( 5 points)
Build a new method that prevents any password from having a similar consecutive
characters.
-------------------------------------------------*/


/**
 * Create the main method
 */
public static void main(String[] args) {

//Declare an instance of Password using the default constructor

// print out the password, using the getter method

// What is the  default password ?.................

// Use the setter method to set a password

// Print out the password using the getter method

/*Declare an instance of the password using the overloaded constructor,
the settings for new password object are:
(2 Capital letters,2 Lowercase letters, 2 Symbols , 2 Numbers)*/

// Set the password using the setter method.

// Print out the password using the getter method


 /*-------------------------------------------------
HW4
Declare an instance of Password to have these settings
(3 capitals, 3 symbols, 3 digits,3 small letters)
this settings should be acceptable, but should be
the maximum and any settings over this should be
an extreme, thus revert back to default settings.
-------------------------------------------------*/

  /*-------------------------------------------------
HW4
Declare an instance of password to have these settings
(0 capitals, 10 symbols, -1 digits,100 small letters)
this settings will be an extreme and thus the user
should get a warning message and the default settings applied.
-------------------------------------------------*/



}
```

```
}
```