Homework 3 (due Wed, 7 Oct) **Homework 5 (due Wed, 21 Oct)**
Homework 4 (due Wed, 14 Oct) **Project 2 (due Wed, 28 Oct)**

The purpose of these assignments is to give you experience with some of the testing techniques we have been discussing in class.

HINT: Each test case will have three passwords as input (one new one and up to two old ones). The software will output ACCEPTED if the new password is accepted and REJECTED if the new password is rejected. Rejected passwords should also indicate why the password was rejected in software output.

Requirements: When checking passwords, use the following requirements:

1. The new password shall be at least 9 characters long, and no longer than 24 characters.
2. New passwords cannot contain any blank spaces, and must contain only numerals, upper- and lower-case letters, and the special characters listed in Requirement 4.
3. All new passwords must contain at least two upper case letters, at least two lower case letters, and at least two numerals.
4. New passwords must contain at least two special characters from the following list:

!	@	#	$	%	&	*	)	(	]	[	}	{	>	<	;	:	.	,	/	|	\	~	?	_	-	+	=

5. New passwords cannot be similar to any of the other two previous passwords, with "similar to" defined as "containing an identical five-character substring (either forward or backward), independent of letter case (for example, `a5%Km` and `a5%kM` would be considered identical substrings).

When developing your test cases, use the techniques discussed in class! This is an iterative process.

1. Use equivalence classes and boundary testing. **HW3:** Explain in detail how you could use these techniques to write test cases for this particular application. Be specific. Apply these techniques and generate at least 8 test cases.
2. Use an orthogonal array. **HW4:** Explain how an orthogonal array could be used to generate test cases that achieve pairwise coverage. Generate the array and show the resulting test cases.
3. Use a decision table. **HW5.** Explain how a decision table could be used to generate test cases. Build a decision table, and then translate each column of the table into a test case.
4. By now you have three sets of test cases, generated using three different techniques. Consolidate them into a single test plan. The maximum number of test cases in this test plan is 30. You may have to combine some overlapping test cases and you may have to leave some out. You can include cases that were not generated in HWs 3, 4, and 5, if you want. ***Your goal is to maximize the chances that your test plan will uncover a bug in a password checker program***. Explain your rationale for keeping the tests you decide to keep.

Automated testing: The password checker program shall read passwords from a file, four lines at a time. The first line of the four lines is a new candidate password; the next two lines of the file represent the user's previous two passwords. The fourth line of the file will be a single upper-case letter in column 1 followed by an optional comment in columns 3-80. Subsequent lines follow this same pattern, in groups of four; each group of four will be checked. Note: *newly accepted passwords do not need to be included in the two "previous" passwords of the next test case*. Your software shall be capable of running a test case manually (the user inputs a new password and two previous passwords), and it shall be capable of reading a file in the previously described format, to determine if your password checker has correctly accepted or rejected all passwords in the test plan.

Here are the milestones:

HW3 – Equivalence class and boundary testing: Due Wednesday, **Oct 7**.
HW4 – Orthogonal array with associated test cases: Due Wednesday, **Oct 14**.
HW5 – Decision table with associated test cases: Due Wednesday, **Oct 21**.

Note – These three HW assignments should all include test cases written in a tabular, Alice-ready format.

Proj2 – Complete test plan with the test cases you decided to keep, expressed as both a tabular test plan ready for Alice and an accompanying file that would run these test cases in automated testing. Due Wednesday, **Oct 28**. **You must have working software by this date**. "Working software" means your software will accept or reject new passwords based on the input and the specified requirements, and is capable of performing automated testing using the specified file format.

When turning in HW assignments, use the following format:

1. **Introduction**. Explain what you are doing. Include any interesting decisions you had to make as a software tester.
2. **Test cases**. These should be in some sort of tabular format that Alice could run. In addition to the input and expected output, each test case should specify which requirement(s) are being tested, and where it came from (orthogonal array, decision table, etc).
3. **Summary** (optional). If you want to reflect on this assignment, to tell more about what you learned, or make observations about how the process is coming along, you can do so here.
4. **Code**. Remember, always put the code at the very end. When I see code, I assume the rest of the document is code.

Remember what the syllabus says about this being an IW class:

It is imperative that your test plans are clear, thorough yet concise, well-organized, and unambiguous. It is not enough to simply have the right test cases – they must be presented in a manner that would facilitate testing for whoever would be running the test plan and reassure the program manager that comprehensive testing has been performed. Overall writing quality will be assessed and be part of your grade for all projects.

This is most pertinent to Project 2; still, the HW assignments should be written in a similarly neat and organized manner.