Advanced Selection & Iteration

Name exercises **[your name] Tutorial xx Exercise yy**, with appropriate xx and yy.

## Exercise 01

1. Design a program that runs continuously in a loop until a Boolean variable becomes false. When the variable becomes false and the loop ends, display an exit message and wait for a final keypress before exiting the program. Use a flowchart to plan the type of loop you want and how control will enter and exit the loop
2. Inside the loop, read from the keyboard and set the loop variable to false if the user enters the word "quit". Make sure you plan and double check this before testing, otherwise you might have an infinite loop. This piece of code, if expanded upon properly, is the right starting point for all later exercises.
3. Test carefully and step through using the debugger to confirm it works properly.

## Exercise 02

Design and write a program that reads integers from the user, tests each of them to see if they are odd or even, and keeps a count of how many are odd and how many are even. After each number is entered, the program should display the latest saying if it is odd or even, and a count displaying how many numbers so far have been odd and even.

## Exercise 03

Design and write a program that reads single characters one at a time from the keyboard and processes them so that:

* Alphabetic characters are displayed in the opposite case, so that 'a' is displayed as 'A' and 'B' is displayed as 'b'.
* Numbers are displayed by subtracting them from 10, so that 9 is displayed as 1 and 4 is displayed as 6.
* The symbols @, ~, # and ! are displayed as they are; all other symbols are ignored/discarded.

Check the program in the debugger to confirm that it works properly, especially when discarding input.

## Exercise 04

1. Design, code and test a program that prompts the user to enter a letter and then prints out whether the letter entered is a vowel, consonant or other.
2. Modify the program so that instead of reading user input repeatedly, the loop runs through the alphabet from 'a' to 'z'. What should change in your code to do this properly?

Can you think of appropriate ways to use constants and/or enumerations in your solution?

## Exercise 05

Design an algorithm that fulfils the following, using a single flow chart:

Using both pseudocode and flowcharts design an algorithm that fulfils the following:

1. Read a date from the user
2. If it is August then I will fly to Egypt. If there is a sandstorm I will stay in the hotel and go for a swim, otherwise I will go for a camel ride, then visit the pyramids.
3. If it is December I will go to the Alps. If there is snow, I will go skiing, otherwise I will eat fondue.
4. At any other time I have to go to class.

This exercise may sound familiar. Go back to your Exercise 01 solution to see how you can improve on your original design, if at all. You will need to add basic user input at various points, e.g. such input 7 for July, 12 for December, 3 for other, and appropriate outputs.

Implement your final design. What type of selection is most appropriate here?

## Exercise 06

An integer variable contains a value of 1 that represents Monday, 3 for Tuesday, 5 for Wednesday, 6 for Thursday, 8 for Friday, 2 for Saturday and 7 for Sunday.

1. Design and write a program using if() statements to input a number into this variable and output "weekday" and "weekend" as appropriate.
2. Design and write the same program using switch statements.
3. Explain in your logbook which you find is the best solution, and why. Can you think of a better solution?

## Exercise 07

Design and write a program that reads single keypresses from the user and displays back the value just read, and which symbol is on the key. Refer to the MicroLesson on **Keyboad Input** if necessary.

## Challenge!

You are to design and code a simple game kind of like random noughts and crosses/tic-tac-toe. Two computer controlled players will take turns randomly picking positions on a game board until there are no more spaces left. A player's pick only puts their symbol on the board if that position is currently empty. When all spaces are filled, the winner has the most positions

1. Design code to draw a 5x5 game board consisting of either 1) blank spaces, 2) Xs or 3) Os.
2. Each time through the game loop, both players pick a random position and try to put their mark there
3. After the moves the board is redrawn and checked for game over/victory.
4. What are the challenging problems? What new information might you need to research?
5. Experiment you can try:
   1. Using the higher number ASCII characters that look like light and dark blocks.
   2. Make the board bigger. How much longer does it take for a game with even a 7x7 board? You want an odd number of squares to avoid draws.
   3. Change the rules so a player can "clear" an opponent's square by picking it.
   4. Do the players alternate turns on each round?

**REMEMBER** - don't forget to copy the finished projects to your personal drive, especially if the project was on the desktop.