C# Exercises – Part 2

# Summary

These exercises are meant to reinforce knowledge from the “Software Engineering Fundamentals in C# - Part 2” PowerPoint.

# Conditionals

## If/else

1. Create an int x and set it to 1
2. Write an if/else if/else statement that will check if x is less than, equal to, or greater than 100 and print an appropriate message for each case
3. Change the value of x to cause each case to be triggered

## Switch

1. Create an enum type. Name it whatever you like and populate it with whatever values you like.
2. Write a switch statement that checks each value of the enum and does something for each match. Include one case that falls through to the next case.

## Order of operations

1. Create a string s and initialize it to whatever you like.
2. Write code that does the following using conditional operators:
   1. If the string length is less than 5 and is odd OR
   2. If the string length is greater than or equal to 10 and is even
   3. Print a message
3. Change the string length to make sure your code works properly

## Nested conditionals

1. Duplicate the logic above using nested conditionals
2. Do not use any conditional operators when comparing values

# Loops

## While

1. Create an infinite while loop and run your program to see what happens
2. Create a string s and initialize it to “0123456789”
3. Use a while loop to print every odd digit in the string, all on the same line.

## Do While

1. Duplicate the above logic using a Do While loop

## For

1. Duplicate the above logic using a for loop

## Foreach

1. Duplicate the above logic using a foreach loop

## Nested loops

1. Imagine you have a hat with 5 different numbers in it. You take out numbers, one at a time, without putting them back.
2. Write code that prints out every possible combination of numbers you can get, in all the different orders you can get them. Use any type of loops you want.

## Loop flow control

1. Declare a string s and initialize it to whatever you want.
   1. Make sure the first character in your string is repeated somewhere else in the string
2. Write a loop that searches through the string and looks for another occurrence of the first character found.
3. If the character found doesn’t match the first one, the loop should immediately go to the next letter
4. If the character found matches, print the index of the match, and immediately exit the loop.

# Arrays

## Single dimensional arrays

1. Create an int array of size 10. Use a while loop to fill the array with the numbers 1 through 10.
2. Use a for loop to print the contents of the array in reverse order.
3. Write code that prints the sum of all numbers in the array
4. Print the contents of the array without using a loop

## Multi-dimensional arrays

1. Create a 2D int array of any size
2. Use any loop(s) to store the product of the two indices in each element
   1. For example, the element at [2,3] should be set to 6
3. Print the contents of the array using a foreach loop

# Functions

## Basic functions

1. Write a function that prints out whatever string you pass into it. Call the function to make sure it works.
2. Write a function that concatenates three strings and returns the result. Call the function to make sure it works.
3. Write a function that takes in an int array, an int index, and an int value. The function should update the array at the specified index with the specified value. If the index is out of range, the function should do nothing. Call the function to make sure it works.
4. Write a function that takes in a string and returns a new string that contains every other word (starting from the first) in the original. Call the function to make sure it works.
   1. For example, passing in “to be or not to be” should return “to or to”
5. Write a function that takes in a string, an optional int parameter called index, and an optional int parameter called length. The function should return a substring of the original string starting at the specified index, with the specified length. Call the function to make sure it works.
   1. If the index or length are out of range, the function should return an empty string.
   2. If the index is not specified, start at index 0
   3. If the length is not specified (or is 0), take the rest of the string

## Reference type equality

1. Write a function called AreArraysEqual that returns bool. The function should take in two int arrays and return whether they contain the same elements, in the same order. Call the function to make sure it works.
   1. The function should return immediately as soon detects a mismatch

## Output parameters

1. Write a function that returns void with two int input parameters. The function should have four output parameters that should return the sum, difference, product, and quotient of the two inputs. Call the function to make sure it works.

## Reference parameters

1. Write a function that takes in a string and a reference to an int. The function should return the character at the index specified by the int.
   1. If the int is negative, the first character should be returned and the int should be changed to 0.
   2. If the int is larger than the string length, the last character should be returned and the int should be changed to the last index in the string.