Logic – Applying Logic to Selection and Iteration

**NOTE** - due to the problems that Visual Studio has with the U: network drive, it will be a good idea to develop and debug your programs on the desktop, or on an external drive or memory stick. In this case please remember to **keep an up to date copy on the U: drive** for me to view at any time, ideally copy it across before the end of the session. If you are using a copy on the desktop be **very** careful that you **do not log off before copying your work back to the U: drive**, otherwise all your work will be **wiped and unrecoverable**. I'd also recommend doing a quick temporary copy to the U: drive several times per session (maybe every 15-20 minutes, or after entering a large section of code), in case there is a power failure or your PC crashes.

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

df

## Exercise 01

Given the following variables:

int i = 16;

int j = 32;

int k = 8;

bool result;

1. Evaluate in your Logbook the following expressions and state the value of **result**:

* result = (i == 16);
* result = (k != 16);
* result = (k < 10);
* result = (j > 33);
* result = (j == (k \* 2));
* result = (j == (i \* 2));
* result = ((k + i) <= (j – i));
* result = ((i / k) == 2);
* result = ((( i + j + k ) / k ) >= 7 );
* result = ((i / k) > (j / i)) ? true : false;

## Exercise 02

1. Write a program that reads in score values for player 1 and player 2, then outputs a message saying which player has the highest score.
2. Add a starting high score defined as 500. If a player score is higher than the high score, display a message saying so, as well as updating the high score.
3. Refer back to your previous exercises for possible solutions.
4. Test your program and confirm that it works using the debugger.

## Exercise 03

1. Referring back to Exercise 01, write a program that **calculates** all of the expressions in and displays them on screen, e.g.: "r = (i == 16) is true"
2. Step through the code using the Debugger to see where unexpected results come from. Use temporary variables if necessary to stores results of individual expressions.
3. Make notes in your Logbook on any surprises.

As cout doesn't display true or false automatically you will have to find out how to do this. Use if() statements to do it , unless you can find another way to do this elegantly. Look back at the tutorials for hints.

## Exercise 04

Examine the following code:

if(false)

;

{

cout << "This text won't be shown." << endl;

}

What do you expect to happen here? Write down what you think before running the program. **COPY AND PASTE** the above code into main() and run the program. Does it match your expectations, if so write down why in your log book, if not find out why and try to explain. Run through it using the debugger to get used to spotting this situation.

## Exercise 05

Examine the following code:

int x = 5;

int y = 7;

if(x = y)

{

cout << "x and y are equal" << endl;

}

What do you expect to happen here? Write down what you think before running the program. **COPY AND PASTE** the above code into main() and run the program. Does it match your expectations, if so write down why in your log book, if not find out why and try to explain. Run through it using the debugger to get used to spotting this situation.

## Exercise 06

Design and implement a program that asks the user for a positive number, then displays a count down from that number down to 1, then displays "FIRE!".

What do you do if the user enters a negative number? Or a string? Adjust your program to handle any user input:

1. By disregarding the input if it's invalid
2. By converting the input to a value number in such a way that the output would make sense to the user.

## Exercise 07

Modify the program from Exercise 06 so that:

1. The countdown does not display even numbers.
2. The countdown is displayed so that a maximum of 10 numbers are displayed on each line.
3. When the countdown goes below ten, instead of display the number in numerical format (5, 4, 3…) it displays the numbers in word format ("five", "four", "three" ..)

## Exercise 08

Design and code a program that reads words entered by the user and checks to see if they are *palindromes* – the same forwards and backwards, like "LOL" or "kayak".

If you can, modify your program so it can read multiple words and check them, like "madam in eden im adam".

## Challenge!

Design and write a number guessing program. Generate a random number within a reasonable range, then ask the user to guess what the number is in a limited number of attempts, e.g., 10. After each guess, display a message telling the user how close the guess was and the number of guesses remaining. Display a final message on win or loss for the user.

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