# Computing I – Programming Labs

## Problem Set 2 - Control Statements

### Problem 1 (Odd or Even)

Write a program involving an `if… else` control statement that reads an integer and determines and prints whether it is odd or even. Output the results as shown below. [Hint: Use the modulus operator. An even number is a multiple of two. Any multiple of two leaves a remainder of zero when divided by 2.]

```
Enter an integer: 0
The integer 0 is even.
```

```
Enter an integer: 1045
The integer 1045 is odd.
```

### Problem 2 (Number Systems Table)

Use a `for` loop to write a program that prints a table of the binary, octal and hexadecimal equivalents of the decimal numbers in the range 1 to 256. You can use the stream manipulators `dec`, `oct` and `hex` to display integers in decimal, octal and hexadecimal formats, respectively. Use tabs (`'\t'`) to display table headers and the numbers, respectively.

Your program output should follow the example shown below.

```
Decimal Octal   Hexadecimal
1       1       1
2       2       2
3       3       3
4       4       4
5       5       5
6       6       6
7       7       7
8       10      8
9       11      9
10      12      a
11      13      b
12      14      c
13      15      d
14      16      e
15      17      f
16      20      10
17      21      11
18      22      12
19      23      13
20      24      14
21      25      15
22      26      16
23      27      17
```

## Problem 3 (Integer Equivalent of a Character, `while` loop)

C++ can represent uppercase letters, lowercase letters and special symbols. You can print a character by enclosing that character in single quotes, as with

```
cout << 'A'; // print an uppercase A
```

You can print the integer equivalent of a character using `static_cast` as follows:

```
cout << static_cast< int >( 'A' ); // print 'A' as an integer
```

This is called a cast operation. When the preceding statement executes, it prints the value 65 (on systems that use the ASCII character set).

Write a program that prints the integer equivalent of a character typed at the keyboard as follows:

- Store the input in a variable `symbol` of type `char` and use the command `cin >> symbol;` to read it in.
- Use a sentinel controlled `while` loop (sentinel `EOF`, `EOF` is `Strg+Z` on Windows).
- In the the `while` loop, test for `EOF` by using `while(!cin.eof())`.
- Use an `if` statement to ignore newline characters (`'\n'`), tabs (`'\t'`), and spaces (`' '`) when printing using `cout`.

Test your program several times using uppercase letters, lowercase letters, digits and special characters (such as #). Follow the screen shots below.

```
Enter a character: A
The integer equivalent of A is 65

Enter a character: Enter a character: a
The integer equivalent of a is 97

Enter a character: Enter a character: 10
The integer equivalent of 1 is 49

Enter a character: The integer equivalent of 0 is 48

Enter a character: Enter a character: #
The integer equivalent of # is 35

Enter a character: Enter a character: ^Z

--------------------------------
Process exited after 57.24 seconds with return value 0
Press any key to continue . . .
```

## Problem 4 (Integer Equivalent of a Character, do… `while` loop)

Rewrite the program developed above using a `do… while` loop. This time, use `cin.get()` to read in user input. Again, use an `if` statement to ignore newline characters (`'\n'`), tabs (`'\t'`), and spaces (`' '`) when printing using `cout`.

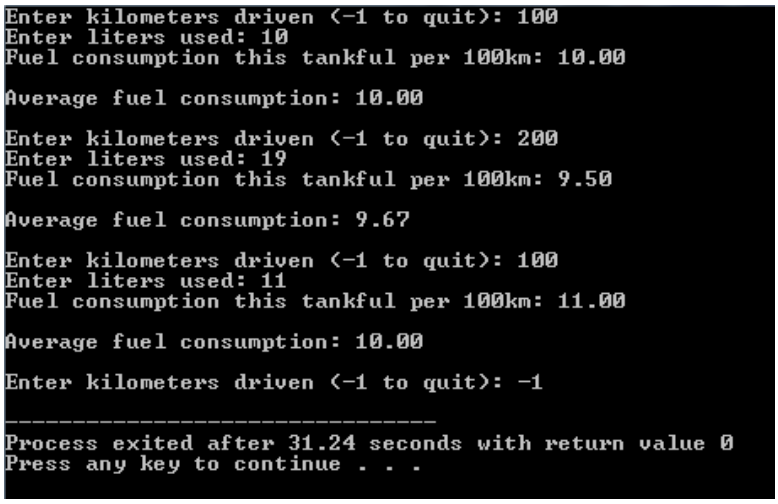
Your output should resemble the screen shot shown below.

```
Enter a character: 100
The integer equivalent of 1 is 49

Enter a character: The integer equivalent of 0 is 48

Enter a character: The integer equivalent of 0 is 48

Enter a character: Enter a character: A
The integer equivalent of A is 65

Enter a character: Enter a character: a
The integer equivalent of a is 97

Enter a character: Enter a character: B
The integer equivalent of B is 66

Enter a character: Enter a character: b
The integer equivalent of b is 98

Enter a character: Enter a character: ?
The integer equivalent of ? is 63

Enter a character: Enter a character: ^Z

--------------------------------
Process exited after 16.57 seconds with return value 0
Press any key to continue . . .
```

## Problem 5 (Gas Mileage)

Drivers are concerned with the mileage obtained by their automobiles. One driver has kept track of several tankfuls of gasoline by recording kilometers driven and liters used for each tankful. Develop a C++ program that uses a sentinel controlled `while` statement (sentinel -1) to input the miles driven and liters used for each tankful. The program should calculate and display the liters per 100km consumed for each tankful and also print the average consumption per 100km for all tankfuls up to this point.

Your program output should follow the example shown below.

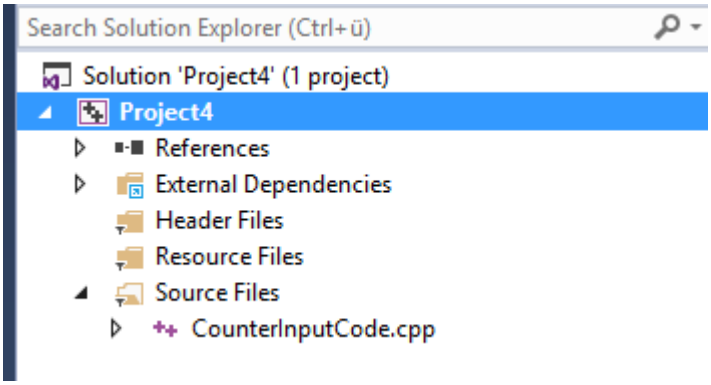


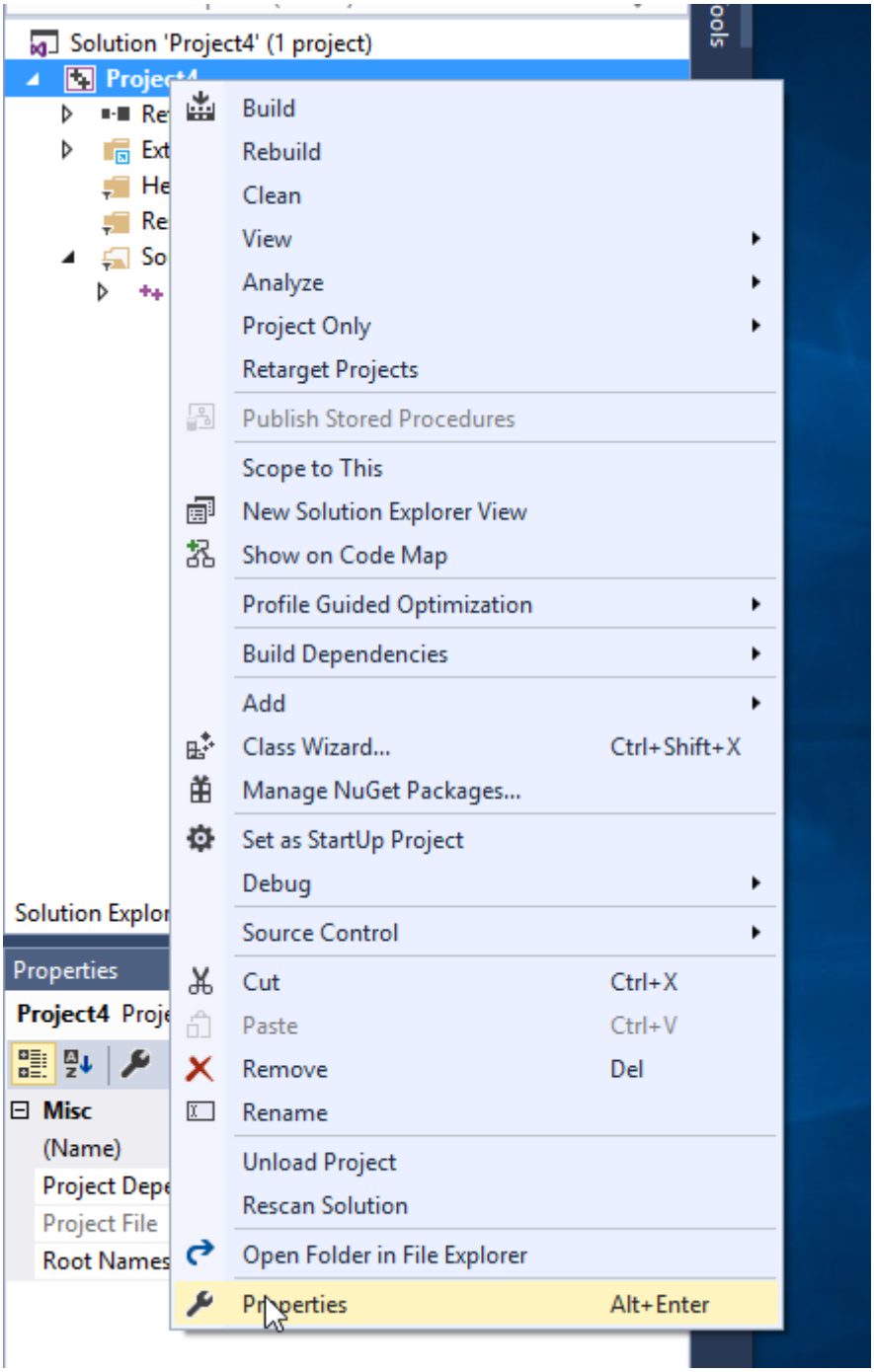
**Note**

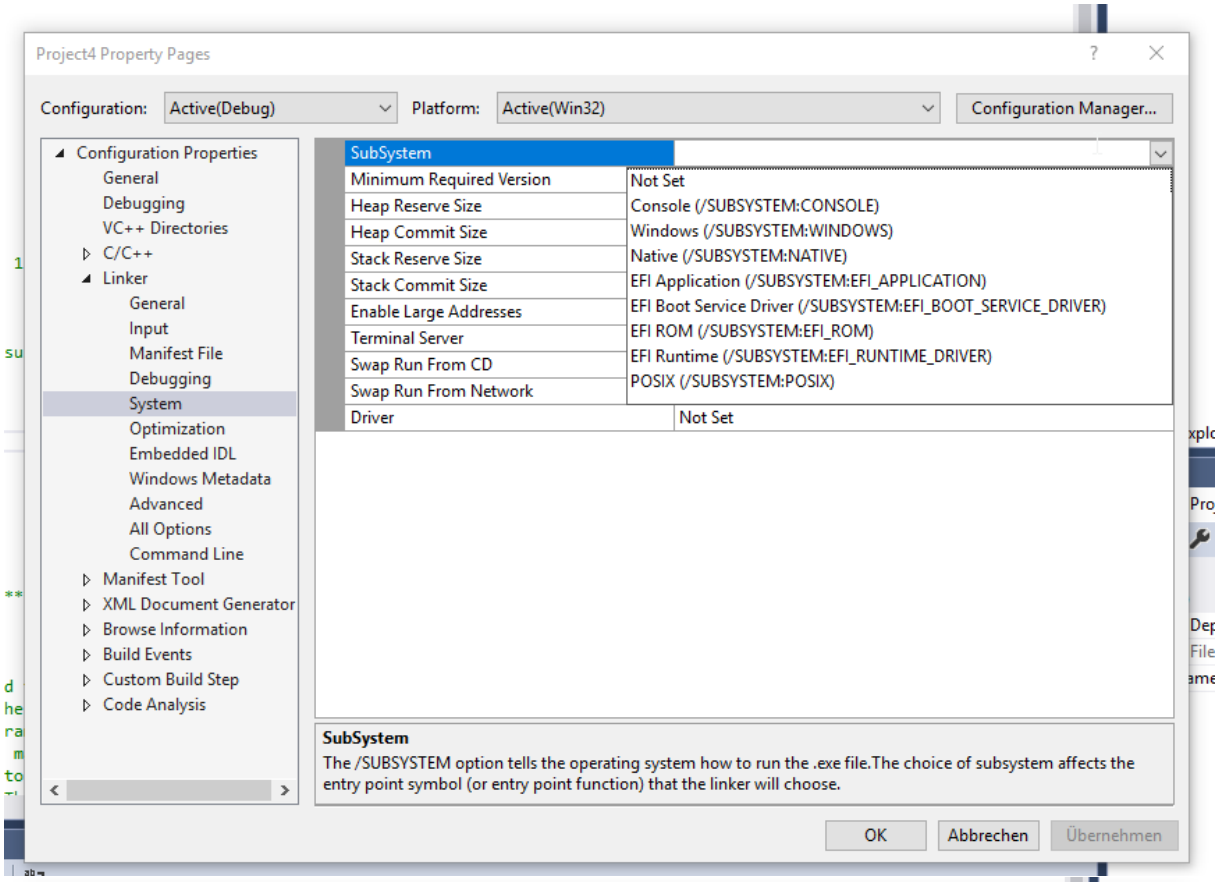
To prevent that Visual Studio closes the console immediately after a program has finished, you need to set up the command **Set Console (/SUBSYSTEM:CONSOLE)** in **Linker Options**. (https://stackoverflow.com/questions/454681/how-to-keep-the-console-window-open-in-visual-c)

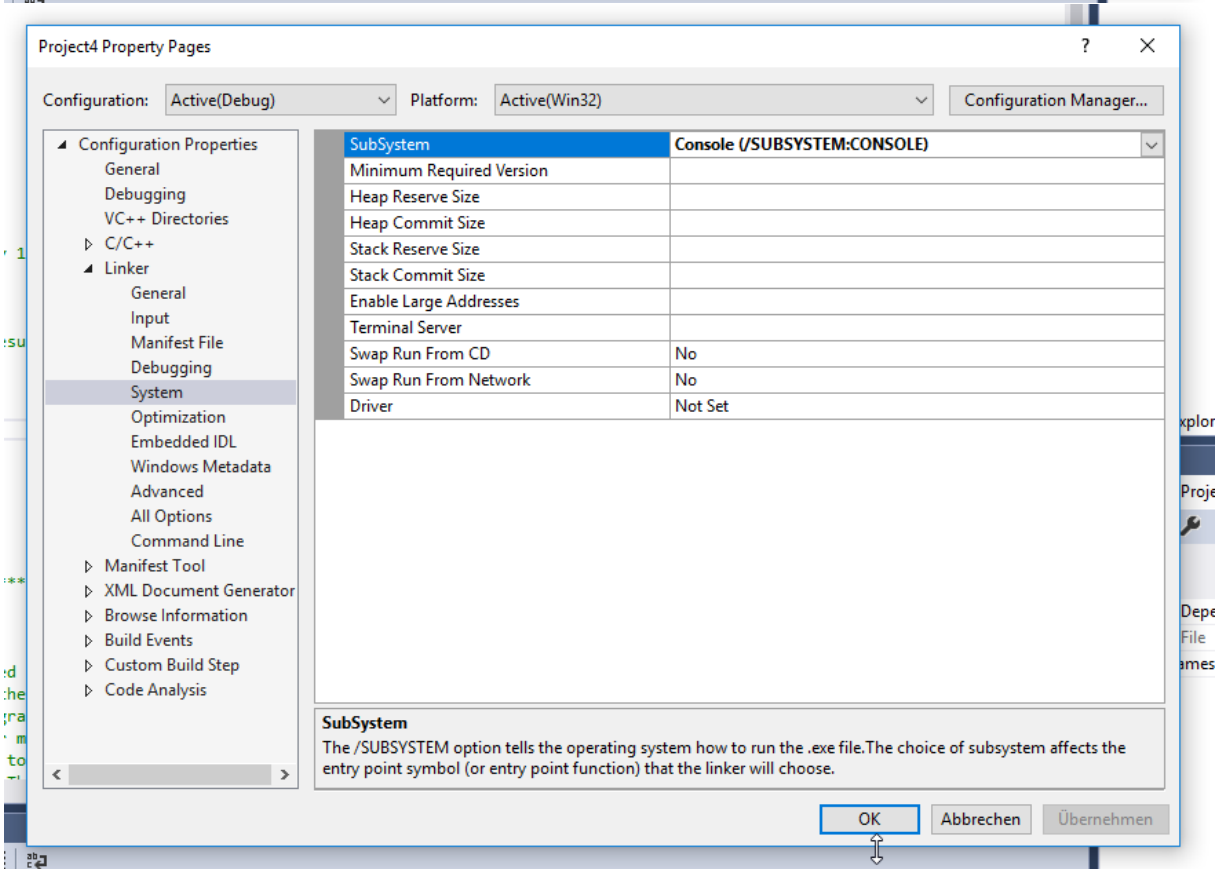
To accomplish this, proceed as follows (see also pictured displayed below):

1. Right-click on project name.
2. Select Properties from context menu.
3. Select Configuration Properties>Linker>System.
4. Click into the drop-down-box to the right and the choose "Console (/SUBSYSTEM:CONSOLE)"
5. Select Apply and then OK.




Solution 'Project4' (1 project)
Build
Rebuild
Clean
View
Analyze
Project Only
Retarget Projects
Publish Stored Procedures
Scope to This
New Solution Explorer View
Show on Code Map
Profile Guided Optimization
Build Dependencies
Add
Class Wizard... Ctrl+Shift+X
Manage NuGet Packages...
Set as StartUp Project
Debug
Source Control
Cut Ctrl+X
Paste Ctrl+V
Remove Del
Rename
Unload Project
Rescan Solution
Open Folder in File Explorer
Properties Alt+Enter
Properties
Misc
(Name)
Project File
Root Names


Project4 Property Pages
Configuration: Active(Debug)
Platform: Active(Win32)
Configuration Manager...
Configuration Properties
General
Debugging
VC++ Directories
C/C++
Linker
General
Input
Manifest File
Debugging
System
Optimization
Embedded IDL
Windows Metadata
Advanced
All Options
Command Line
Manifest Tool
XML Document Generator
Browse Information
Build Events
Custom Build Step
Code Analysis
SubSystem
Minimum Required Version
Heap Reserve Size
Heap Commit Size
Stack Reserve Size
Stack Commit Size
Enable Large Addresses
Terminal Server
Swap Run From CD
Swap Run From Network
Driver
Not Set
Console (/SUBSYSTEM:CONSOLE)
Windows (/SUBSYSTEM:WINDOWS)
Native (/SUBSYSTEM:NATIVE)
EFI Application (/SUBSYSTEM:EFI_APPLICATION)
EFI Boot Service Driver (/SUBSYSTEM:EFI_BOOT_SERVICE_DRIVER)
EFI ROM (/SUBSYSTEM:EFI_ROM)
EFI Runtime (/SUBSYSTEM:EFI_RUNTIME_DRIVER)
POSIX (/SUBSYSTEM:POSIX)
SubSystem
The /SUBSYSTEM option tells the operating system how to run the .exe file.The choice of subsystem affects the entry point symbol (or entry point function) that the linker will choose.
OK
Abbrechen
Übernehmen


Project4 Property Pages
Configuration: Active(Debug)
Platform: Active(Win32)
Configuration Manager...
Configuration Properties
General
Debugging
VC++ Directories
C/C++
Linker
General
Input
Manifest File
Debugging
System
Optimization
Embedded IDL
Windows Metadata
Advanced
All Options
Command Line
Manifest Tool
XML Document Generator
Browse Information
Build Events
Custom Build Step
Code Analysis
SubSystem
Console (/SUBSYSTEM:CONSOLE)
Minimum Required Version
Heap Reserve Size
Heap Commit Size
Stack Reserve Size
Stack Commit Size
Enable Large Addresses
Terminal Server
Swap Run From CD
No
Swap Run From Network
No
Driver
Not Set
SubSystem
The /SUBSYSTEM option tells the operating system how to run the .exe file.The choice of subsystem affects the entry point symbol (or entry point function) that the linker will choose.
OK
Abbrechen
Übernehmen