

## ACOM1900 – Introduction to Programming Control flow Worksheet 2007/8

In each of the following you may assume that the values entered by the user are always of the expected *type*. You should not assume that they are in the expected *range*.

1. Write a program that prompts the user to enter a number and displays whether or not that number is positive (we will assume that zero is a positive number).
2. Write a program that prompts the user to enter two numbers and then displays the result of dividing the first by the second. Be sure to handle the case where the second number entered is zero! For example:

```
tony@gwydyon:~$ ./divide.py
Enter a number: 4
Enter another number: 3
4 divided by 3 is 1.3333333333
tony@gwydyon:~$ ./divide.py
Enter a number: 4
Enter another number: 0
Cannot divide by zero!
```

3. Write a program that prints the 7 times table. Print the first 13 terms. If your program is longer than four lines, it's wrong!
4. Write a program that prompts the user to enter a string and which then displays the string with one character on each line.
5. Write a program that prompts the user to enter a string consisting of digits. The program should examine the string and then display how many 1s, 2s and so on were found. For bonus marks, the program should also display how many characters in the string were not digits.

The output might resemble:

```
tony@gwydyon:~$ ./digits.py
Enter a string of digits: 0113 343 5768
Analysis of 0113 343 5768:
0s: 1
1s: 2
2s: 0
3s: 3
```

4s: 1  
5s: 1  
6s: 1  
7s: 1  
8s: 1  
9s: 0

2 characters were not digits.  
tony@gwydyon:~\$ ./digits.py  
Enter a string of digits: 1123581321  
Analysis of 1123581321:  
0s: 0  
1s: 4  
2s: 2  
3s: 2  
4s: 0  
5s: 1  
6s: 0  
7s: 0  
8s: 1  
9s: 0

All characters were digits.