Introduction to Testing

Testing is …

When we write computer programs, it's really important that we check to make sure the programs are working correctly. We've already talked about "debugging", that is where we find out that the program has an error in it when we try to run it, and then have to fix it, so "testing" is where we carefully examine the program to check if it is working correctly, and to check if it is doing what we want it to do.

Testing Inputs

If programs take in values from users, it's important to test those programs on a variety of different inputs to see how they react, and if they can detect the difference between valid and invalid inputs. So, I usually try:

Expected Input	Test Values
Number	I usually start off with a few small numbers (1, 2, 3), then I try a few bigger numbers (99999, 999999), then I try zero (0) often programs don't consider a zero input, and then I'd try a few negative numbers (-1, -2, -3, -9999999). Next, I'd try a few characters (A, B, C, @), and finally I'll try the <Space> character.
Characters	I'd try some uppercase letters (A, B, C), then I'll try some lowercase letters (a, b, c), then I'll try some numeric characters (1, 2, 3), and a few other characters (@, #, %). Following these I'd try a few strings of characters (AAAA, BBBB, CCCC, @@@@), and finally I'll try the <Space> character.
Date	I try today's date, then twenty years ago, then I try a few simple valid dates (10/10/1010, 11/11/1111), and some invalid dates (22/22/2222, 00/00/0000) and then the <Space> character.

Testing Outputs

We can also check if our programs give the right outputs, so, for example, if we write a program to double a number, but instead of multiplying the input number by 2, we accidently multiplied it by 3, our program wouldn't give an error, but it's still wrong:

```
# PROGRAM DoubleNumber:
OurValue = int(input("Please input value: "))
print("Double that number is", OurValue * 3)
# END.
```

So, if we typed in the number 4, we know we're expecting 8, but we'd get the wrong answer, 12. It's really important to check programs to see if they are consistently giving the correct answer by having a set of known inputs and expected outputs.

Reflections

These are two simple examples of testing (checking inputs and output), but testing includes a wide range of approaches to check if a program is working correctly.