Practical Sheet 2

Using Python for Binary Fun

Aims

Understand how Python handles numbers in different bases and consider whether using Python helps to teach binary.

Section		Aim
1	Then Concept of a Literal	Writing numbers in programs
2	Literals in Different Bases	Writing numbers in bases other than decimal
3	Numbers to Strings	How to ask Python to convert number to its representation in different bases
4	Strings to Numbers	How to ask Python to convert strings in different bases to numbers

Related topics

- ***Topic 2.1 Binary Representation***
- ***Topic 2.2 Converting Binary and Decimal***

Read about how Python handles different number bases. Try the examples using Python as you go along.

1 The Concept of 'Literals'

When you type 5.5, +5.5 or -2 in a program, you are using a 'literal'. The complete rules for literals are quite complex (for example, they include ways to enter letters that are not in the Roman alphabet). Here are some of the forms of literal Python understands:

- 5
- -5.7
- +10
- 1.1e-2
- "String on first line. \nContinued on second line"

Exercise 1.1

Which of the following are valid Python literals (find out by trying them out).

```
+ 20      +20.00    -20e3     7.89e-3
```

2 Literals in Different Bases

Python allows numbers to be entered in bases other than base 10 – either binary, octal or hexadecimal. All these numbers start with a ZERO and then have a letter:

Letter	Base
b or B	Binary
o or O	Octal
x or X	Hexadecimal

Remember: '0xABC' is digit 0, letter 'x', letter 'A'. This is particularly confusing in Octal: e.g. 0o77 is an octal number: digit 0, letter o, digit 7, digit 7.

Here is an interactive Python session:

```
>>> 0x77
119
>>> 0o77
63
>>> 77
77
>>> 0b1001101
77
```

Numbers are entered in different bases but are always echoed in decimal. Try this out. You will see that you can use it to check the binary sums above. For example, try:

```
0b11 + 0b10
```

Exercise 2.1

Choose a number and enter it in different bases.

Exercise 2.2

Which numbers are represented in the same way in all bases?

3 Numbers to Strings

Python includes functions to convert numbers to strings in different bases.

Function	Base	Example	Result
hex()	Convert integer to hexadecimal string	`hex(99)`	`'0x63'`
oct()	Convert integer to octal string	`oct(99)`	`'0o143'`
bin()	Convert integer to binary string	`bin(99)`	`'0b1100011'`
str()	Convert integer to a decimal string	`str(99)`	`'99'`

Note that the result is a **string.**

The 'str' function the default. It is often used implicitly (for example in the print function). Occasionally, you need it explicitly.

Exercise 3.1

Try the examples in the table and add some similar examples.

4 Strings to Numbers

We used the int() function to convert a string (e.g. a string typed in by the user when the 'input' function is used) to an integer. Base ten is the default, but we can use any other base. The base we want to use is given after the **string** we are converting.

Here are some examples:

```
>>> int('101', 2)
5
>>> int('101', 8)
65
>>> int('101', 10)
101
>>> int('101', 16)
257
>>> int('101', 23)
530
```

The last one is obviously weird, but it's a computer, so it does weird without complaining.

Exercise 4.1: Try some similar examples.

5 Further Exercises

Exercise 5.1

Try the following examples and explain what is happening:

```
>>> 0b101 + 0b011

>>> 0o65 + 0x65

>>> hex(0b1111)

>>> oct(0xAB)

>>> hex(int('110', 2) + int('110', 8))
```

Exercise 5.2

Write a binary conversion program. The user types a binary number and the computer prints in out in different bases. Here is an example with the user input underlined:

```
Enter a binary number> 101010
In base 10, the number is: 42
In octal, the number is: 0o52
In hexadecimal, the number: 0x2a
```