Getting Started With Turing – Primitive Data Types

The basic unit of information on a computer is called a *bit*, which has two states: *on* or *off* (1 or 0). Bits are generally grouped into units of 8, which is called a *byte*. A single byte can store all of the possible combinations of the 8 bits, or $2^8 = 256$ values.

In Turing, as with most programming languages, there are various *data types*, which are used to represent and (as variables) store different types of data. Each data type has different demands on how much space it requires in the computer's memory, and this is measured in bytes.

Character Data Types

A single character would be any letter, number, or symbol that you can type on your keyboard. This even includes special keys like the <Enter> and <Backspace> buttons, but we won't explore those for now. Use single quotes to represent a character:

'A' '$' '+' '7'

Integer Data Types

Recall: Integers are whole numbers (no decimals), positive or negative, including zero. In mathematics, we would represent the integers using a number line:

...	-3	-2	-1	0	1	2	3	...

Mathematically, the integers stretch an infinite distance in the positive and negative direction. For computer systems, there are limits on the size of the numbers that can be stored. In Turing, this limit is **4 bytes**, which defines how much space in memory is used for a single integer value.

Type	Size (bits)	Size (bytes)	Range	Approximate Range
int	32	4	-2^{31} to $2^{31}-1$	$\pm 2\ 000\ 000\ 000$

Integer constants must not be written with a decimal point, and they cannot contain any separators between digits.

For example, each of the following is an illegal integer value.

(a) 37.0 contains a decimal point
(b) -12 562 contains a blank space between digits
(c) 1,233,985 contains commas between digits

Real Data Types

For very large integer values, or values with decimals, another data type is required. It stores data in a completely different way, using scientific notation and powers of 10.

Value	as integer	in scientific notation	text representation
1 trillion	1 000 000 000 000	1×10^{12}	1e12

As you can see in this example, one trillion is too large to be an integer value (as integers can only represent up to about two billion). With scientific notation, however, the number is actually represented by two different numbers: 1 and 12. The computer stores this large number in a form similar to the text representation, where the 'e' stands for "10 to the exponent".

Type	Size (bits)	Size (bytes)	Precision	Approximate Range
float	32	4	at least 6 decimal places	$\pm3.4\times10^{38}$

Each of the following is a valid real constant.

5.23 .3 2818. -0.0002 6.7E5

Real constants can also be written in a form similar to that used in scientific notation.

(a) 5.6e2 represents the value 5.6×10^{2} giving $5.6\times100=560$
(b) 37E-4 represents the value 37×10^{-4} giving $37\times0.0001=0.0037$
(c) -0.667e-2 represents the value -0.667×10^{-2} giving $-0.667\times0.01=-0.00667$
(d) 3E2.5 is illegal, because the exponent contains a decimal

Boolean Data Type

The **`boolean`** type represents a *truth* value. There are only two boolean values: `true` or `false`. The words true and false are reserved and cannot be used for any other purpose. Boolean values are very compact, requiring only 1 bit for storage.

String Data Type

Although the string data type is fairly simple to use in Turing, it is actually a complicated structure. We will discuss it in detail at some point in the future.

Exercises

1. Classify each constant as integer, real, or illegal. If it is illegal, justify your answer.
   (a) -0.0123
   (b) .73
   (c) 5e7
   (d) 5E6.2
   (e) 33,146
   (f) -6E-7
   (g) -0
   (h) $ 7.21
   (i) +45.2
   (j) 8.29e

2. State, with reasons, which of the following are not legal integer constants.
   (a) -47
   (b) 23.
   (c) -0
   (d) 22 900

3. Rewrite in standard decimal form.
   (a) 2.94e1
   (b) 0.0004e3
   (c) -2e-3
   (d) 26.77e-3
   (e) -54E-3
   (f) -.3e1