Introduction to Programming in Turing

Formatted Output

Formatted Output in Turing

To have more control over the appearance of output, you may wish to format your output. Here is some unformatted output.

```
put "This" ..
put "is" ..
put "a" ..
put "test!!!"
```

Output:

```
Thisisatest!!!
123412112345 67
```

Formatted Output in Turing

With formatting, it is possible to specify how many spaces with be required for each output. In this case, each output will use at least 5 spaces (notice the last output required more than 5, so it took 7).

```
put "This" : 5 ..
put "is" : 5 ..
put "a" : 5..
put "test!!!" : 5
```

Output:

```
This is    a     test!!!
1234512345123451234567
```

Formatted Output in Turing

For numbers, it generally works the same way. The difference is that some numbers (real) may have decimals. It is also possible to use formatting to specify the number of decimal places.

put <number> : <minimum spaces> : <decimal places>

```
put 1 : 5 : 2
put 22222.2 :5: 2
put 3.1415 : 5 : 2
put 4.9999: 5 : 2
```

In this example, for each output we have specified at least 5 spaces wide, and 2 decimal places for each.

Formatted Output in Turing

put <number> : <minimum spaces> : <decimal places>

```
put 1 : 5 : 2
put 22222.2 :5: 2
put 3.1415 : 5 : 2
put 4.9999: 5 : 2
```

Most of these numbers only needed 4 spaces (1 leading digit, 1 decimal point, and 2 decimal places). The 5th space was put at the beginning.

Output:

```
 1.00
22222.20
 3.14
 5.00
12345
```

One number needed 5 leading digits, 1 decimal point, and 2 decimal places, so it exceeded the specified 5 spaces.

Formatted Output in Turing

put <number> : <minimum spaces> : <decimal places>

```
put 1 : 0 : 2
put 22222.2 :0: 2
put 3.1415 : 0 : 2
put 4.9999: 0 : 2
```

Output:

```
1.00
22222.20
3.14
5.00
```

Unless you are doing some very specific formatting, you just want the number rounded. In that case, use zero for the first number, and specify decimals using the second number.

Formatted Output in Turing

This also works when you have multiple items on the same line, and when using variables containing numbers.

```
var x : real
x := 2.718

put "x is ", x : 0 : 2
```

In this example, the output will be:

```
x is 2.72
```