Introduction to Python

Strings

Components of Python

1) Variables (Me hold data)

2) Data-type (integers, float,)

3) Data-structures (list, tuple, dicitonary)

4) Loop (for, while)

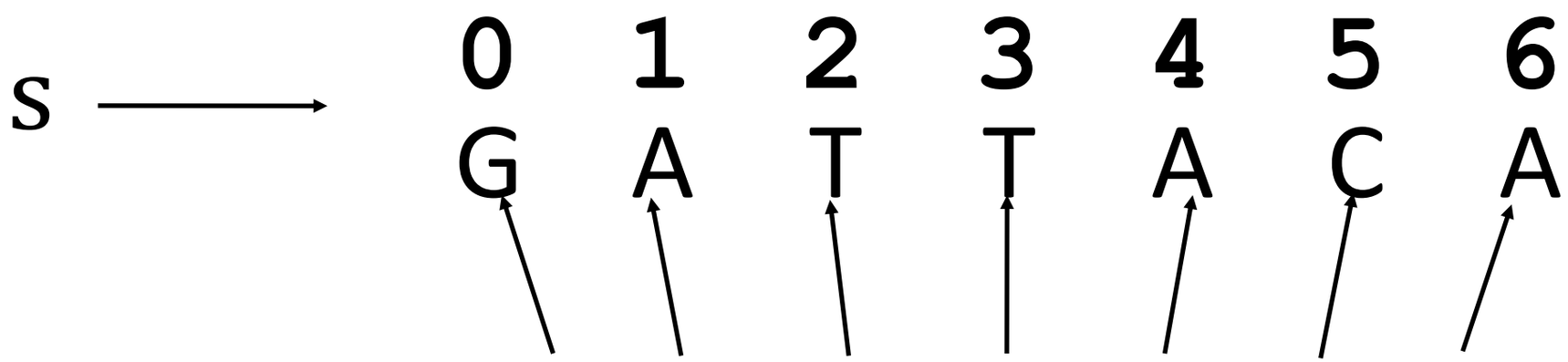
5) Branch (if, elif, else)

6) Methods

7) Functions

Computers store text as strings

```
>>> s = "GATTACA"
```



's' is variable!

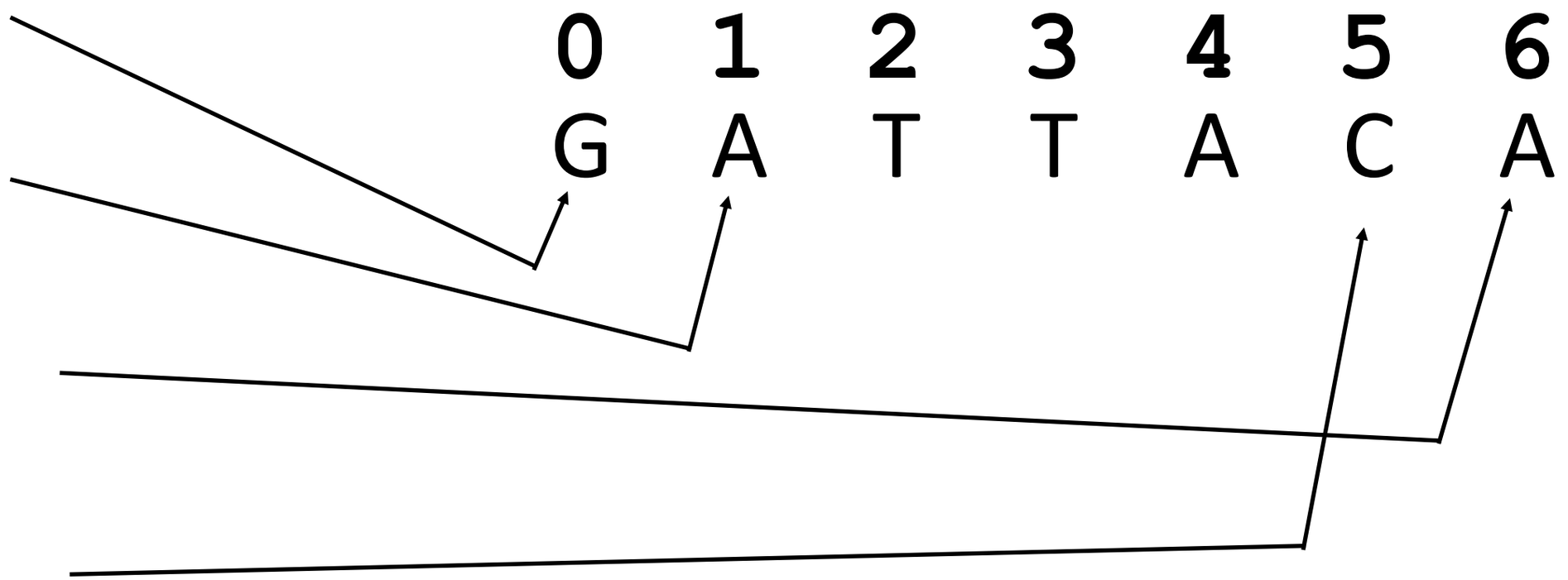
Each of these are characters

Why are strings important?

- Sequences are strings
  - `..catgaaggaa ccacagccca gagcaccaag ggctatccat..`
- Database records contain strings
  - `LOCUS AC005138`
  - `DEFINITION  Homo sapiens chromosome 17, clone hRPK.261_A_13, complete sequence`
  - `AUTHORS  Birren,B., Fasman,K., Linton,L., Nusbaum,C. and Lander,E.`
- HTML is one (big) string

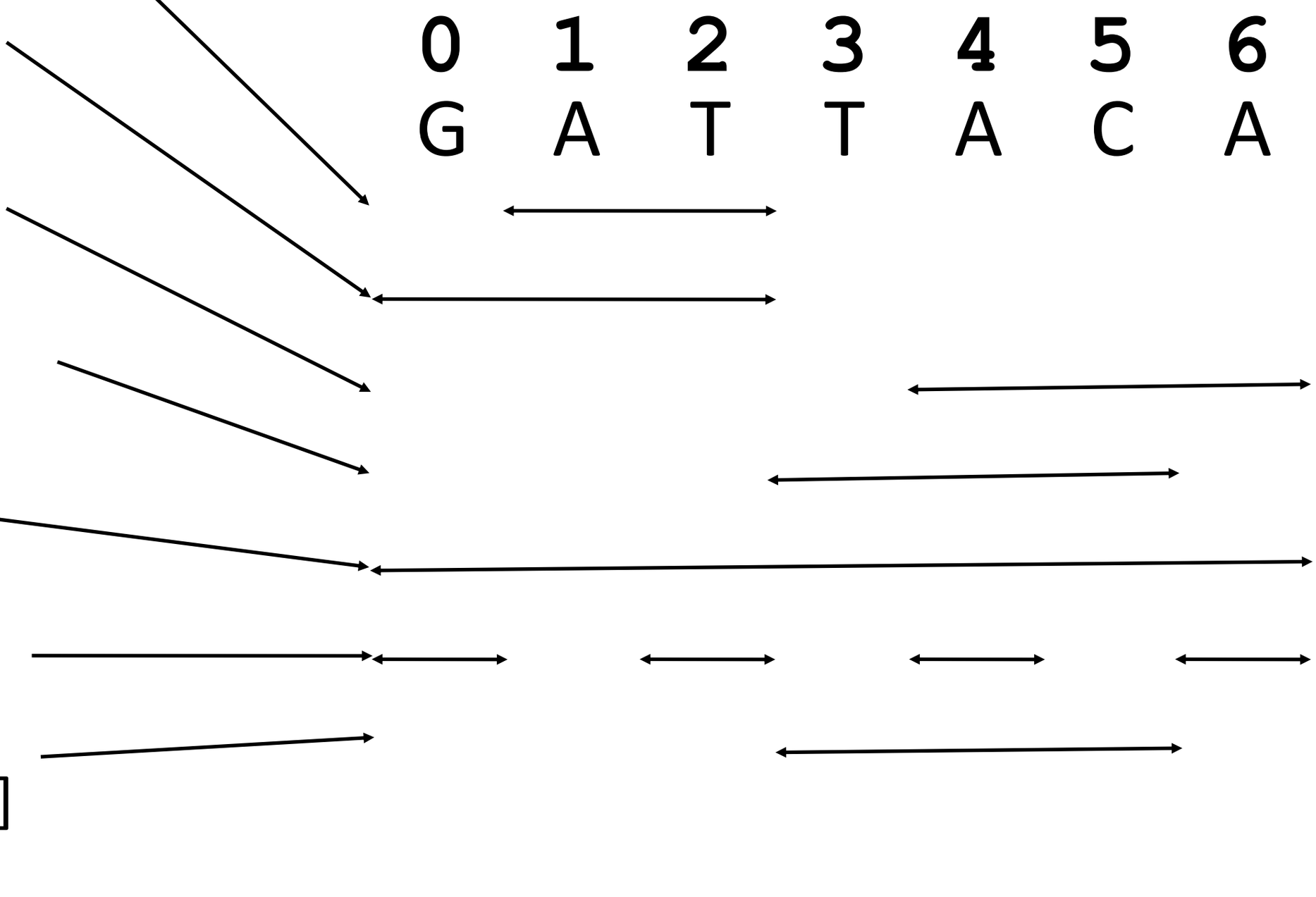
Getting Characters

```
>>> s = "GATTACA"
>>> s[0]
'G'
>>> s[1]
'A'
>>> s[-1]
'A'
>>> s[-2]
'C'
>>> s[7]
Traceback (most recent call last):
  File "<stdin>", line 1, in ?
IndexError: string index out of range
>>>
```



Getting substrings

```
>>> s[1:3]
'AT'
>>> s[:3]
'GAT'
>>> s[4:]
'ACA'
>>> s[3:5]
'TA'
>>> s[:]
'GATTACA'
>>> s[::2]
'GTAA'
>>> s[-2:2:-1]
'CAT'
```



Creating strings

Strings start and end with a single or double quote characters (they must be the same)

"This is a string"
"This is another string"
""
"Strings can be in double quotes"
‘Or in single quotes.’
'There’s no difference.'
‘Okay, there\’s a small one.’

Special Characters and Escape Sequences

Backslashes (\) are used to introduce special characters

```
>>> s = 'Okay, there\'s a small one.'
```

The \ “escapes” the following single quote

```
>>> print s
Okay, there's a small one.
```

Some special characters

Escape Sequence	Meaning
\\	Backslash (keep a \)
\'	Single quote (keeps the ')
\"	Double quote (keeps the ")
\n	Newline
\t	Tab

Working with strings

```
>>> len("GATTACA")
7
>>> "GAT" + "TACA"
'GATTACA'
>>> "A" * 10
'AAAAAAAAAA'
>>> "G" in "GATTACA"
True
>>> "GAT" in "GATTACA"
True
>>> "AGT" in "GATTACA"
False
>>> "GATTACA".find("ATT")
1
>>> "GATTACA".count("T")
2
>>>
```

length

concatenation

repeat

substring test

substring location

substring count

Converting from/to strings

```
>>> "38" + 5
Traceback (most recent call last):
  File "<stdin>", line 1, in ?
TypeError: cannot concatenate 'str' and 'int' objects
>>> int("38") + 5
43
>>> "38" + str(5)
'385'
>>> int("38"), str(5)
(38, '5')
>>> int("2.71828")
Traceback (most recent call last):
  File "<stdin>", line 1, in ?
ValueError: invalid literal for int(): 2.71828
>>> float("2.71828")
2.71828
>>>
```

Change a string?

Strings cannot be modified
They are immutable
Instead, create a new one

```
>>> s = "GATTACA"
>>> s[3] = "C"
Traceback (most recent call last):
  File "<stdin>", line 1, in ?
TypeError: object doesn't support item
assignment
>>> s = s[:3] + "C" + s[4:]
>>> s
'GATCACA'
>>>
```

Some more methods

```
>>> "GATTACA".lower()
'gattaca'
>>> "gattaca".upper()
'GATTACA'
>>> "GATTACA".replace("G", "U")
'UATTACA'
>>> "GATTACA".replace("C", "U")
'GATTAUA'
>>> "GATTACA".replace("AT", "**")
'G**TACA'
>>> "GATTACA".startswith("G")
True
>>> "GATTACA".startswith("g")
False
>>>
```

Ask for a string

The Python function “raw_input” asks the user (that’s you!) for a string

```
>>> seq = raw_input("Enter a DNA sequence: ")
Enter a DNA sequence: ATGTATTGCATATCGT
>>> seq.count("A")
4
>>> print "There are", seq.count("T"), "thymines"
There are 7 thymines
>>> "ATA" in seq
True
>>> substr = raw_input("Enter a subsequence to find: ")
Enter a subsequence to find: GCA
>>> substr in seq
True
>>>
```

Assignment 1

Ask the user for a sequence then print its length

```
Enter a sequence: ATTAC
It is 5 bases long
```

Assignment 2

Modify the program so it also prints the number of A, T, C, and G characters in the sequence

```
Enter a sequence: ATTAC
It is 5 bases long
adenine: 2
thymine: 2
cytosine: 1
guanine: 0
```

Assignment 3

Modify the program to allow both lower-case and upper-case characters in the sequence

```
Enter a sequence: ATTgtc
It is 6 bases long
adenine: 1
thymine: 3
cytosine: 1
guanine: 1
```

Assignment 4

Modify the program to print the number of unknown characters in the sequence

```
Enter a sequence: ATTU*gtc
It is 8 bases long
adenine: 1
thymine: 3
cytosine: 1
guanine: 1
unknown: 2
```