

Comparing strings

How can we compute a boolean for (“apple” < “banana”)?

Strings are compared lexicographically

e.g. left to right, comparing each corresponding pair of letters

Each letter is represented as a number on the computer


ascii encoding (man ascii): ex: ‘A’ = 65, ‘B’ = 66, etc

every letter corresponds to a positive integer

`ord(c)` ← returns ascii value for character

`chr(num)` ← returns character for an ascii value

encoding refers to how numbers map to characters -> ascii isn't the only one!



Exercise - Ascii values

What is the result of “apple” < “APPLE”? Explain in terms of ascii values.

What is the ord of “A”, “0”, or “ “?

Check that “apple” == “apple” is True

Check that “apple” == “Apple” is False

Check that “apple” == “apple ” is False (**watch out for the extra space!**)

Dot Dash

Approach 1

```
dot_dash1.py -- Atom
File Edit View Selection Find Packages Help
quit.py dot_dash1.py
1 """
2 Prompt the user for a string and then prints a symbol
3 between each letter based on the length. If the length
4 of the string is even, put a dot ("."); Otherwise,
5 put a dash ("-")
6
7 The solution uses nested for loops in an if statement
8 """
9
10 def main():
11     word = input("Enter a string: ")
12
13     word_length = len(word)
14     if word_length % 2 == 0: # even
15         result = "."
16         for i in range(word_length):
17             result = result + word[i] + "."
18     else:
19         result = "-"
20         for i in range(word_length):
21             result = result + word[i] + "-"
22
23     print(result)
24
25 main()
26
dot_dash1.py 17:1 LF N UTF-8 Python 0 files
```

Analysis:

How does the accumulator work on lines 15-17?

word = cats

word_length = 4

result = "." # initial value

Iteration	i	word[i]	result = result + word[i] + "."
1	0	"c"	result = "." + "c" + "." = ".c."
2	1	"a"	result = ".c." + "a" + "." = ".c.a."
3	2	"t"	result = ".c.a." + "t" + "." = ".c.a.t."
4	3	"s"	result = ".c.a.t." + "s" + "." = ".c.a.t.s."

Dot Dash

Approach 2

```
dot_dash2.py ~ - Atom
File Edit View Selection Find Packages Help
quit.py dot_dash2.py
1 """
2 Prompt the user for a string and then prints a symbol
3 between each letter based on the length. If the length
4 of the string is even, put a dot ("."); Otherwise,
5 put a dash ("-")
6
7 The solution uses nested if statements in a for loop
8 """
9
10 def main():
11     word = input("Enter a string: ")
12
13     word_length = len(word)
14
15     if word_length % 2 == 0: # even
16         result = "."
17     else:
18         result = "-"
19
20     for i in range(word_length):
21         if word_length % 2 == 0: # even
22             result = result + word[i] + "."
23         else:
24             result = result + word[i] + "-"
25
26     print(result)
27
28 main()
dot_dash2.py 13:28 LF N UTF-8 Python 0 files
```

Dot Dash

Approach 3

```
dot_dash3.py ~ Atom
File Edit View Selection Find Packages Help
quit.py dot_dash4.py dot_dash3.py
1 """
2 Prompt the user for a string and then prints a symbol
3 between each letter based on the length. If the length
4 of the string is even, put a dot ("."); Otherwise,
5 put a dash ("-")
6
7 This solution uses a delimiter variable
8 """
9
10 def main():
11     word = input("Enter a string: ")
12
13     word_length = len(word)
14
15     if word_length % 2 == 0: # even
16         delimiter = "."
17     else:
18         delimiter = "-"
19
20     result = delimiter
21     for i in range(word_length):
22         result = result + word[i] + delimiter
23
24     print(result)
25
26 main()
27
dot_dash3.py 16:24 LF N UTF-8 Python 0 files
```