AI showcase: Summerization

Task: Check the online versions of various newspapers to identify current topics. Write one summary article per topic.

http://newsblaster.cs.columbia.edu/

Examples of if-statements in Python

simple:

```
if x==”bye” :
   print “Bye, bye!”
   print “Nice talking to
                     you.”
```

two-way:

```
if x > y :
   return x
else :
   return y
```

multi-way:

```
if x > y :
   print “You win!”
elif x < y :
   print “I win!”
else :
   print “It's a draw!”
```

Conditions

Conditions are expressions that evaluate to `True` or `False`, i.e., expressions that create an object of type **boolean.**

Some comparison operators:

==	equal to
!=	not equal to
>	greater than
<	less than
>=	greater or equal to
<=	less than or equal to

work for numbers and strings!

More complex conditions

if it *rains* **or** *snows* **and** I **don't** *have an umbrella …*

Boolean operators: `and, or, not`

```
if x>y

if x>y and y>z

if not(x>y and y>z)

if not(x>y and y>z) or x<z
```

Boolean algebra: Complex Conditions

and

True	and	True	=> True
True	and	False	=> False
False	and	True	=> False
False	and	False	=> False

or

True	or	True	=> True
True	or	False	=> True
False	or	True	=> True
False	or	False	=> False

not

not	True	=> False
not	False	=> True

Practice using if-statements in Python

- Implement a function that finds the greatest of three numbers. Don't use the built-in `max` function.
- Many companies pay time-and-a-half for any hours worked above 40 in a given week. Write a function that takes the number of hours worked and the hourly rate and calculates the total wages for the week.
- A person is eligible to be a US senator if they are at least 30 years old and have been a US citizen for at least 9 years. To be a US representative these numbers are 25 and 7, respectively. Write a program that asks for a person's age and years of citizenship as input and outputs their eligibility for the Senat and House.

loops

... carry out some instructions repeatedly

go 10 steps north

go north until you see the wall

go north as long as you don't see the wall

while-loops

```
while some condition is true:
    do this
```

Examples:

```
while you don't see the wall to the north:
    go north
```

```
ask the user for input and read it
while the user input is not "quit":
    print what the user typed
    ask the user for more input
```

```
let x = 0
while x is not equal to 10:
    print x
    x = x + 1
```

Practice using while-loops

- Write an algorithm that sums up all numbers from 0 to 100
- Write an algorithm that reads in numbers from the user and calculates the average of those numbers. The algorithm should keep reading in numbers until the user says that he has no more numbers.
- Write an algorithm that reads in numbers from the user and sums them up until the sum is greater than 1000.

while-loops in Python

```
while <condition>:
    <statements>
```

```
x = 0
while x != 10:
   print x
   x = x + 1
```

```
x = raw_input("Type sth: ")
while x != "quit":
   print x
   x = raw_input("Type more: ")
```

Practice using while-loops in Python

- Write a program that reads in numbers from the user and sums them up until the sum is greater than 1000. It then prints out the sum and the average of those numbers.
- Another number guessing game: The computer randomly generates a number. Then it asks the user to input a number and tells the user whether this number is smaller or greater than the randomly generated number. The user gets to guess again and gets feedback again, and this is repeated until the user guesses the right number.
- The Fibonacci sequence starts 1, 1, 2, 3, 5, 8, ... Each number in the sequence (after the first two) is the sum of the previous two. Write a program that computes and outputs the nth Fibonacci number, where n is a number entered by the user.

Essential computer architecture schema

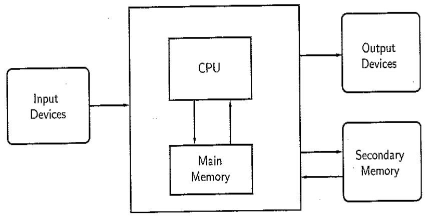

Figure 1.1: Functional View of a Computer.

Practice Working with Strings

- Write a function that takes a string as parameter and replaces all occurrences of *a* by *o*.
- Write a function that takes a string and replaces all vowels by *o*.
- Make up five keywords. Write a function that takes a string and checks whether one of these five keywords appears in the string. Have the function print out the first keyword that it finds in the string.
- Do the same as before but have the function print out all of the keywords that occur in the string.
- Write a function that takes a string and encodes it into ASCII. I.e., for every letter in the string it prints out its ASCII number. (Hint: You have to use a loop to go through the string letter by letter.)

Practice Pattern Matching

Test your patterns by trying to match it on various strings (re.search(<pattern>, <string>)).

- Write a pattern finds all variants of the verb *like* in a string, i.e., *liked, likes, liking.*
- Write a pattern that matches any of the following sequences:

Paul likes … .	*Paul hates … .*
Mary likes … .	*Mary likes … .*
Mary doesn't like … .	*Mary doesn't like … .*
Paul doesn't like … .	*Paul doesn't like … .*

More Pattern Matching Exercises

- Write a function that takes a string and then checks whether it matches the paul-mary-like-hate pattern that you defined previously. If it does print a response in the following style:

paul likes pizza	=>	*mary hates pizza*
mary hates pizza	=>	*paul likes pizza*
mary does not like pizza	=>	*paul likes pizza*

- Think of an inputer-response-patterns that you remember from Eliza. Write a function that takes a string as parameter, checks whether this pattern matches, and if so, outputs the appropriate response.