INTRO TO PROGRAMING (08/26/2019)

DAY 3. HISTORY OF COMPUTER

Goal: think about how to solve problems with computers

History of computers:

* 1000s years ago: people counting things
* 1600s-1700s: small machines that can help with math ideas
* 1800s: theory is getting better
* 1930s/1940s: “almost” computers. Moto f them could do just one task
* 1950s…..1970s: constant advancements
* 1980s: personal computers

Constantiing changing today.

How does a computer work?

>>>5+5

10

5+5 needs to be translated by the computer:

* Assembly language: add 5,5 which means 5+5. Some people use it.
* Machine language: CPU Works with binary numbers. People rarely use it

-Abstraction: Phyton provides a high-level (easy) abstraction over the computer.

-Only

Why are there so many computer languages?

- Different tradeoffs. Phyton is very easy to use, you lose some expressiveness at low levels (ex. of tradeoff of Phyton).

Will we créate programs with Window interfaces?

- Most programs we use will be text-based.

When we write programs?

They are like a recipe or series of steps. The computer does not “think” for itself: the programs tell it what to do.

Natural vs artificial languages

Syntax: - rules for constructing a ”correct” statement

- Computers are very strict on syntax.

- It is hard to fully describe.

Semantics: - “meaning” or “does this actually do my task”

DAY 5. TURTLE FOLLOW-UP

Grading/rubric

Joe.getscreen()

Joe.getscreen().screensize()

Pyton terminology

Joe=turtle.Turtle(): Joe is the variable

Identifier is some name you give to something

Quick note\*: When you make a variable joe is not the same as Joe/JOE

We name variable in lower snake\_case. Ex:my\_favorite\_number=6

Turtle method: is a code that acts on a variable or object: joe.goto(x,y). In this case “goto…” is the method; in other woords is “calling” a method. When you call a method you need ALWAYS ().

The things inside the () (x,y) are called parameters/arguments. They comunícate between our code and the method. Allows the method to have another results each time. You have to fill them in order, or give names (x=…,y=…). The opposite would be position. Defaulted parameter: when the definition includes a default value, you do not need to supply it. The opposite parameter would be required

Colors

Black: code

Orange: special key Word

Purple: built-in ítem

Red: error or comment

Green: indicates a “string” (text that is meant to be text, not variable, method or parameter.

Tips for coding

Use blank lines to make readable

Do not forget comments

Style guide

Creativity

Execution: does it work?

*When it Works, go back and make a code nice*

DAY 6. PHYTON BASIS

NameError: name “” is not defined

We have to do first import “” and we create the feature

The print operation will make things show up on the output

Built-in functions are purple in INDLE

Noun: kind of like variables or data

Verb: method or function

Variable names start with a letter, lowercase. We prefer descreptive names over 1-letter names. Multiple words are separated by \_ (number\_of\_sizes)

x= 7 \* math.sqrt(14 - 3/2) / math.sin(19)

Variables and values in Python have “types”. We will use 4 frequently:

1. Integer (regular number): 7,1000,-45…
2. Float (decimal point number):1.5,-0.075…
3. Boolean (true, false)
4. String (words).

If you need a “ inside the string , then use ´ on the outsides

If you need a ‘ inside the string, then use “ on the outsides

If you need both “ and ‘ inside a string, then escape (\) one or the other

“Collections”: is posible to combine multiple of those other types into one larger collection. Ex: (x,y) you can coordenate in one variable.

Interactions between the types and operators:

Sometimes are not-compatible with some operators.

Every time you have a variable keep in your head what type it is

DAY 7. Converters

def: definition

Every time we type a colon, the very next line will be indented. In Python, the colon is a little bit special, and it directs the flow of the program

int: ask the user for input an returns it to the program for later use

The most common thing we will use int is to convert a string into a number

str()

float()

print(): display results to users

Everything on the very left edge of the file will be run inmediately, but ‘def’ statements don’t actually do anything unless you call that function

DAY 12.

Patterns are very common and general.

Give a name to the pattern:

We increase t

Container of multiple items:

“Container types”: LIST🡪 collection of items [12 , 7 “hello” , True]

A LIST contains multiple items

DAY 13: LISTS

Names = [ ] empty list

Names = [“joe”]

Names = [ “joe”, “bill”]

Names = [ [“hello”], 7.4]

How to add a third name?

List are changeable:

Names.append(“fred”) 🡪 add item at end of existing list

Names.inse(number, item) 🡪 adds item at specific index

Names.pop(2) 🡪 removes and returns item at specific index. Counting starts at 0 for index

Names.remove(“joe”) 🡪 removes an item by value

Names.sort() 🡪 sorts the names in order

Sorted(names) 🡪 temporary sort; it doesn’t mess up the original order

Look in Phyton Tutorial (5.1) and Python Documentation for more information

DAY 14.

Exam I Friday 4 October

Random selection:

input(random)

random.random() # number from 0.0 up to 1.0

random.randrange(int)

random.choice(list) # choose 1 item out of the list and return it

Input validation:

You need to handle unexpected errors

* Inside a function
* Only return when data is good (if statement)
* Often loop, frequently while True

DAY 15: MIDTERM EXAM

We wanna check 2 things:

If response = rock

If response = ‘r’

F or T 🡪 T

F or F 🡪 F

Common error: response == “rock” or “r”

Exam (75-100 points total):

* Multiple choice (15-20):

Ask a question or show some code

* Written answer (5-10)

More substantial question or longer piece of code

Our response may have to be written code or just describe a solution

Study python keywords + builtins and operator chart

DAY 16:

Local variable: A variable that lives in the function and only exists there.

Global variable: A variable that is accessible everywhere in the program.

Only “def” should be on the left border of your program.

CONSTANT: a variable that holds value that never changes and is assigned at the every start of your program. Ex: conversion factor