Why did the computer show up late to work?

It had a hard drive!

Source: https://www.rd.com/jokes/computer/

Thank you Colton!

CMPT 120

Lecture 6 – Chatbots
Robustness -> User Validation, Efficiency, Testing (Step 4 of Software Development process) and Errors
Feedback – Assignment 0

• Thank you for all your jokes!
• Lots of great jokes!
• Some of you took the opportunity to practise the Python building blocks we have learnt so far!

• Make sure you satisfy the requirements
  • Write a Python program that outputs a computer joke to the computer monitor screen when it is executed/run.
    • This means: use print( ... ) function
    • Not a computer joke
  • Your program must also print the source of your computer joke, i.e., the link or location where you found the joke.
    • Do not put your source in a comment

• Reminder:
  • For Assignment 1: there are no extension given
    • This means that you have to submit your program on time.
Careful!

- This is on the Shell:

```python
Python 3.12.1 (v3.12.1:2305ca5144, Dec  7 2023, 17:23:39) [Clang 13.0.0 (clang-1300.0.29.30)] on darwin
Type "help", "copyright", "credits" or "license()" for more information.
>>> # Assignment0.py
>>> #
>>> # Description: Python program that outputs a computer joke to the computer monitor screen when it is executed/run.
>>> #
>>> # Author: 
>>> # Date: W Jan. 17 2024
>>> #
>>> # Ask the user if they want to hear a joke
>>> input("Would you like to hear a joke?"
Would you like to hear a joke?

>>> input("Why did the computer keep sneezing? It had a virus!"
Why did the computer keep sneezing? It had a virus!
```

- It is not a Python program created using the Editor!

- Make sure you submit the right program! 😊
Last Lecture

✓ We continued practicing using conditional statements in our Python programs
  ✓ What if there are many conditions (many branches)?
  ✓ What if we are dealing with integers?
  ✓ Can these conditional statements be nested?

✓ We also played around Boolean values and Boolean expressions
Let’s finish this one first!

• **Step 1 - Problem Statement**
  • Write a grade-to-letter grade converter that converts a grade into letter grade.
Today’s Menu

• Improving grade-to-letter grade converter
  • Robustness -> User input Validation
  • Efficiency

• Step 4 Testing and Errors

• Our Guessing Game:

Your turn!

• Step 1 - Problem Statement
  • Write a guessing game, which allows a user to guess a number between 1 and 10.
Let’s practice a little!

What does this output if the user types **kale**?

```python
salad = input("What salad do you want to eat? ")
if salad == "lettuce" or salad == "kale":
    print("That's healthy.")
elif salad == "kale":
    print("That's great.")
else:
    print("Woo!")
```
How about this one?

What does this output if the user types **kale**?

```python
salad = input("What salad do you want to eat? ")
if salad == "lettuce" or salad == "kale":
    print("That's healthy.")
elif salad == "kale":
    print("That's great.")
else:
    print("Woo!")
```
Hand Tracing

• What is it?
  • When a software developer manually goes through her/his code (program) and “execute” it as if s/he was a computer, mimicking the Python Interpreter

• Why doing it?
  • To figure out what our program does/produces, hence to verify whether our program is solving the problem
  • To determine whether our program contains any errors
Robustness

- **strongly** formed or constructed
  - Merriam Webster
- able to **withstand** or **overcome adverse conditions**
  - Oxford Dictionary

- What if the user enters a grade $< 0$ or $> 100$?
- **User Input Validation**
Efficiency

• Consider this axis:

• Let’s go to our Python code!
More efficient? How?

Original version:
```python
# If grade is between 80 and 89 (inclusively), the letter is B
elif grade >= 80 and grade <= 89 :
    print("B")
```

versus

improved version:
```python
# If grade is between 80 and 89 (inclusively), the letter B
elif grade >= 80:
    print("B")
```
Step 4 Testing

• Syntax error
  • Example: `print(int("23bottles"))`

• Runtime error
  • Example: `print(int("23bottles"))`

• Semantic error

• When you test your code
  • A test case is made of:
    1. Test data
      • Data - must be specific
      • We need to choose this data before we execute our program
    2. Expected result
      • The result we expect our program to produce with this data
      • We need to compute it before execute our program
    3. Actual result
      • The result our program actually produced (and printed on the screen?)
      • Our program passes the test if `expected result = actual result`
  • How many test cases must we create?
Last Lecture - Your turn!

- **Step 1 - Problem Statement**
  - Write a guessing game, which allows a user to guess a number between 1 and 10.
Step 4 – Testing

• Testing our guessing game:
  1. Test case 1: input != number to guess
  2. Test case 2: input == number to guess

• How to know the number to guess?
  • The trick is to ...
Robustness - User Validation

• What if the user enters a guess < 1 or > 10
• We know how to deal with this situation!
  • Testing our new version of our guessing game:
    1. Test case 1: input != number to guess
    2. Test case 2: input == number to guess
    3. Test case 3: invalid input: 53 (> 10)
    4. Test case 4: invalid input: -21 (< 1)
Robustness - User Validation

• What if the user enters "banana"?
  • Misbehaving user versus well-behaved user
    • Testing our new version of our guessing game:
      1. Test case 1: input != number to guess
         • How to know the number to guess?
           • The trick is to …
      2. Test case 2: input == number to guess
      3. Test case 3: invalid input: 53 (> 10)
      4. Test case 4: invalid input: -21 (< 1)
      5. Test case 5: invalid input: "banana"
Review: How to construct a Boolean condition?

Some Python functions return numerical values
Here are some examples:
• `len("hello")` that returns 5,
• `int("27")` that returns 27
• "hello".find("lo") that returns 3

Similarly, there are Python functions that return Boolean values
Here are some examples:
• `all([1<2,2<4,5==5])` that returns `True`,
• "123456".isdigit() that returns `True`,
• "123456".isalpha() that returns `False`

**SYNTAX:** .<function>(...) <function>(...) operator

Example:
<string>.isdigit() etc...
Example:
`all(...)` etc...
Example: the `in` containment test operator etc...

Result of **Boolean expression:** `True` or `False`
Summary

• Feedback from Assignment 0
• Improving grade-to-letter grade converter
  • Robustness -> User input Validation
  • Efficiency
• Step 4 Testing and Errors
• Our Guessing Game:

Your turn!

• Step 1 - Problem Statement
  • Write a guessing game, which allows a user to guess a number between 1 and 10.
Next Lecture

- Let’s see how much we have learnt so far by having our first **Practice Exam**!
- Great chance for us …
  - To hone your software development skills
  - To become familiar with:
    - Types of questions asked in CMPT 120 exams
    - Writing code on paper
    - To work in teams
    - And to ask all your questions!
- Our first **in-class activity** -> 1%
  - I will ask you to hand in your answer to one of the questions in our Practice Exam #1

Course grading scheme on our course website: **Best 7 in-class exercises out of 10: 1% each, for a total of 7%**