Lecture 16: Introduction to Computer Programming Course - CS1010

DEPARTMENT OF COMPUTER SCIENCE | 03/18/2019

Rensselaer

Announcements

- Homework 7 is due tonight
- No Homework posted this week!
- Exam 2 is on Thursday March 21

Goals for today

- Errors
- Exception Handling
- Exam 2 Review

Common Errors

- Syntax errors/parsing errors, are perhaps the most common kind of errors you get while you are still learning Python:
  - while True print('Hello world')
  - ^
  - SyntaxError: invalid syntax
- The error was caused by the missing colon and that is what the little arrow depicts.

Errors

- Bound to happen in programs.
- One common cause is to use a code block in an unexpected way
- One technique used to deal with errors is called exception handling
- Under this technique the code will run for the rest of the script and error handling will handle error.

Exception

- Even if a statement or expression is syntactically correct, it may cause an error when an attempt is made to execute it.
- Errors detected during execution are called *exceptions* and are not unconditionally fatal:
  - You will soon learn how to handle them in Python programs.

Exceptions

- An exception is an error that happens during execution of a program.
- When a particular error occurs, Python generates an exception that can be handled, which avoids your program to stop.
- When you are aware that you have a code which can produce an error then you can use **exception handling**.
- You can **raise an exception** in your own program by using the raise exception statement.
- Raising an exception **breaks** current code execution and **returns the exception** back until it is handled.

Some common errors

- **IOError**
  - If the file cannot be opened.
- **ImportError**
  - If python cannot find the module
- **ValueError**
  - Raised when a built-in operation or function receives an argument that has the right type but an inappropriate value
- **KeyboardInterrupt**
  - Raised when the user hits the interrupt key (normally Control-C or Delete)
- **EOFError**
  - Raised when one of the built-in functions (input()) hits an end-of-file condition (EOF) without reading any data
- **IndexError**
  - Raised when index out of range

Key words

- To catch exceptions following keywords are used:
  - **try**: This means the block of code to be attempted that might lead to error
  - **except**: The block of code to be executed when there is an error in the try block
  - **finally**: The final block of code to be executed regardless of error

Syntax

- Basic Syntax :
- try:
- // Code
- except:
- // Code

Try and except

- To use exception handling in Python, you first need to have a catch-all except clause.
- The words "try" and "except" are Python keywords and are used to catch exceptions.
- try-except blocks
  - The code within the try clause will be executed statement by statement.
  - If an exception occurs, the rest of the try block will be skipped and the except clause will be executed.

Else Statement

- In python, you can also use else statement with try-except block which must be present after all the except clauses.
- The code enters the else block only if the try clause does not raise an exception.
- Let us check in spyder

Exam Details

- Everything from Lecture 9 to Lecture 15
- Use the In-Class Study Material and Homeworks.
- You can bring 2 A4 sized handwritten sheets (both sides)
- It is a closed book, closed computer exam.
- Today’s Lecture Material (Errors and Exception handling) NOT covered in Exam 2.

Structure

- 6 Questions worth 100 points.
- Question 1: What is the output of a given program?
  - Will have 4-5 parts.

Example: What is the output of the following:

```
for num in range(5):
    for i in range(num):
        print (num, end=" ")
print("\n")
```

1

2 2

3 3 3

4 4 4 4

Structure

- Q2 will ask you to write a program.
  - Review loops
- Q3 Has 5 parts that ask you to write what a particular line of code does. For example what does this mean:
  - L1=[1,2,3]
  - Answer: Creates a list L1 of size 3.
- Q4. Given some code point out the error if there is any.
- Q5. Operations on lists, for example l1= [‘a’,’b’,’c’], find the index of element ‘c’. There are 5 parts.
- Q6. Has 3 parts each asking to write some code involving loops. You are free to use while or for loops OR even no-loops (if it works)

Review problems for exam (Problem 1)

- What is the output of this program:
- Number = 5
- for i in range(1, Number):
- for j in range(1, i + 1):
- print(‘*’, end=' ')
- print("")

Problem 2

- What does the following code do:
- A. tuple4 = ('rpics',)*3

  print(tuple4)
- B. im = Image.open(filename)

  im.rotate(90)

Problem 3

- Find the error in the following code:
- Hint: *Look for errors that stop the program from running*
- def join_strings(x):
- return x[1] + x[2]

- join_strings([“New", “Problem"])

Problem 4: To demonstrate loops

- Given a list: animals = ['cat', 'monkey', 'hawk', 'tiger', 'parrot']
- Capitalize all names in the list.

Problem 5

- Consider co2_levels in a city for 11 days = [320.03, 322.16, 328.07, 333.91, 341.47, 348.92, 357.29, 363.77, 371.51, 382.47, 392.95]
- Pretend sum() function does not exist, find the average of the Co2 levels for 11 days.

More on Exam 2

- Practice problems where:
  - You are asked to find the index of elements in a list.
  - Do all list operations: indexing and Slicing
  - Access elements of a list using loops.
- Review all problems from Class Exercise 7 onwards
- Everything is posted on Submitty under the folders:
  - Code
  - Class_Exercises

Check the full list of exceptions

- https://docs.python.org/3/library/exceptions.html
- https://docs.python.org/2/library/exceptions.html#exception-hierarchy