Intro to Programming (No Prior Experience)

Finals Review Part I

Emily Zhao
T/R 4:55PM–6:10PM

Modules 1 – 10 will be covered on the final

- Module #1 (Variables, Statements, etc.)
- Module #2 (Types, Operators, Debugging)
- Module #3 (Boolean Logic, Using Modules)
- Module #4 (While Loops)
- Module #5 (For Loops, Nested Loops)
- Module #6 (Functions)
- Module #7 (Strings, Sequences, Slicing)
- Module #8 (Lists)
- Module #9 (Exceptions, Input/Output)
- Module #10 (Dictionaries)

Math Expressions

- Math operators (+, -, /, //, *)
- Writing math expressions
- Evaluating math expressions
- Storing & printing the results of math expressions
- Difference between the two division operators (/ and //)
- Order of operations in math expressions
- The exponent operator (**)
- The modulo operator (%)

First-half Material

Second-half Material

Data Types

- What is a data type?
- Strings
- Numeric data types
  - Integers (int)
  - Floating point numbers (float)
- Mixed type expressions
- Data type conversion
  - Using the float() and int() function to convert strings into numbers
  - User input & data types (converting strings to floats / ints for calculation purposes)
- The Boolean data type
- Boolean variables

Output with the print() function

- General use of the print function and its default behavior
  - Unlimited arguments
  - Spaces inserted between arguments
  - Line break after each call to the function
- Customizing line endings (end=”)
- Customizing argument separators (sep=”)
- Escape characters (\n, \t, etc.)

Basic String Manipulation

- Combining two strings (concatenation) – "+" operator
- Multiplying a string (repetition) – "*" operator
- Formatting numbers using the format() function
  - Formatting Strings – width, left align, right align, center align
  - Formatting Integers – width, left align, right align, center align
  - Formatting Floats – width, left align, right align, center align, # of decimal places, "," separator
- Case manipulation using str.lower() and str.upper()
- Calculating string length using the len() function

Selection Statements

- The structure of an IF statement (IF keyword, condition, colon, indentation)
- Writing a condition for an IF statement
- Boolean operators (<, >, ==, !=, >=, <=)
- Comparing numeric values using Boolean expressions
- Comparing string values using Boolean expressions
- Using the IF-ELSE statement
- Nesting decision structures (IF statements inside other IF statements)
- The IF-ELIF-ELSE statement
- Logical operators (and, or, not)

Condition Controlled Loops

- The structure of a "while" loop
- Mechanics & how they work
- Setting up conditions for a while loop
- Infinite loops and how to work with them
- Sentinels (defining a value that the user enters that causes the loop to end)
- Input validation loops (asking the user to continually enter a value until that value matches some condition)
- Setting up and using accumulator variables
- Self referential assignment statements (i.e. counter = counter + 1)
- Augmented assignment operators (i.e. counter += 1)

The Range Function

- mechanics and how the function works
- creating simple ranges (i.e. range(5))
- creating ranges with defined start and end points (i.e. range(3,10))
- creating ranges with a step value (i.e. range(5,50,5))
- creating ranges that count backwards (i.e. range(50,5,-5))
- user controlled ranges (i.e. range(1, somevariable))

Functions

- mechanics and how functions work
- function definitions
- arguments
- return values
- calling a function
- local variables (variables that are defined inside a function and can only be accessed inside that function)
- passing arguments to your own functions
- passing multiple arguments to your own functions
- global variables (variables created outside a function that can be accessed by any part of your program)
- making changes to global variables inside a function using the ‘global’ keyword
- writing a value returning function (i.e. using the ‘return’ keyword to send a result from your function to the part of your program that called that function)
- returning multiple values from a function
- Input, Processing & Output notation

Miscellaneous Concepts

- Generating random numbers
- Errors & error types
- Debugging strategies
- Pseudocoding

Modules

- Creating a module
- Defining functions in a module
- Calling functions in a module

Exceptions

- Preventing exceptions using selection statements (i.e. using an “if” statement to prevent an error from occurring)
- Using the try / except / else suite to test problematic code for an error and “catch” it before it has a chance to crash your program.

Lists

- Simple Variables vs. Lists (simple variables can only hold one piece of data, but lists can hold multiple values) – you can think of a list like a “book” and a variable like a “sheet of paper”
- Defining lists in Python (i.e. mylist = [1,2,3])
- Concatenating lists with the “+” operator
- Repeating lists with the “*” operator
- Referencing list items using index notation (i.e. mylist[0])
- Iterating through a list using a “while” loop
- Iterating through a list using a “for” loop
- Using the len() function to determine the # of items in a list
- Updating the value of an item in a list using bracket notation
- Creating empty lists

Lists (cont'd)

- Finding an item in a list using the "in" operator
- Adding items to a list using the append method
- Sorting items in a list using the sort method
- Reversing items in a list using the reverse method
- Finding the position of an item in a list using the index method
- Inserting an item in a list at a specific index using the insert method
- Finding the largest and smallest values in a list using the min and max methods
- Totaling the values of all elements in a list using an accumulator variable
- Removing an item from a list using the remove method
- Storing lists in files
- Reading lists from files using the readlines method

Strings Manipulation

- Iterating through all characters in a string using a for loop
- Indexing a specific character in a string using bracket notation
- Iterating through all characters in a string using a while loop
- String immutability (you can’t change a string using bracket notation like you would change a list element)
- Testing a string for substrings using the “in” operator
- Detecting character types in a string using the built-in string testing methods (isdigit, isalpha, isalnum, islower, isupper, isspace)
- Splitting a string into a list using the “split” method

File Input & Output

- Opening a file for writing
- Opening a file for reading
- Writing data to a file
- Delimiters (separating data in a file)
- Reading data from a file using the read() method
- Reading data from a file into a list
- Processing data stored in a file

Dictionaries

- What is a Dictionary?
- Basic usage of Dictionaries
- Differences between Lists and Dictionaries
- Knowing when to use which data type (lists for numerically indexed data, dictionaries for string-indexed data)

Object Oriented Programming

- What is a class?
- Setting up instance variables
- Accessing instance variables through 'dot syntax'
- Constructor functions
- Mutability of instance variables
- Instance methods
- The 'self' keyword

Review Questions

Write code to generate the following output:

```
begin
K$M$TL N P
B#
A RW@Oend
```

Write code to generate the following output:

```
begin
K$M$TL N P
B#
A RW@Oend
```

```
print("begin")
print("K", "M", "T", sep="$", end="")
print("L", "N", "P")
print("B#")
print("A", "R", end="")
print("W", "O", sep="@", end="")
print("end")
```

What are the datatypes of the following?

```
format(605.35, '.2f')
input("Enter an integer: ")
data.split(",")
"Emily".upper()
file.read()
```

What are the datatypes of the following?

```
format(605.35, '.2f')
input("Enter an integer: ")
data.split(",")
"Emily".upper()
file.read()
```

```
→ string
→ string
→ list
→ string
→ string
```

Fill in the blanks to yield the correct output

```
num1 = "10"
num2 = _____ ( ___(num1)___ , ____ )
_____ ( ___ , ____ )
```

```
20.0012345#
```

Fill in the blanks to yield the correct output

```
num1 = "10"
num2 = _____ ( ___(num1)___ , ____ )
_____ ( ___ , ____ )
```

```
num1 = "10"
num2 = format(int(num1)*2, "<10.2f")
print(num2, end="#")
```

```
20.0012345#
```

What’s the output?

```
"Professor".append("Zhao")
["Grace"] +  "Hopper"
"Madness" - "ness"
"Potato"[2]
["Hello"] + ["World"]
```

What’s the output?

```
"Professor".append("Zhao")
["Grace"] +  "Hopper"
"Madness" - "ness"
"Potato"[2]
["Hello"] + ["World"]
```

```
→ Error
→ Error
→ Error
→ t
→ ["Hello", "World"]
```

Describe what the following function is doing:

```
nums = []

for i in range(5, 10):
    total = 0
    for j in range(i):
        total += j
    if total % 2 == 0:
        nums.append(i)

print(nums)
```

Describe what the following function is doing:

```
nums = []

for i in range(5, 10):
    total = 0
    for j in range(i):
        total += j
    if total % 2 == 0:
        nums.append(i)

print(nums)
```

```
# For the numbers 5 through 9
# checks to see if the sum of the digits
# from 1 to itself is even
# If it's even, adds to nums list
```

Write the for loops as while loops

```
nums = []

for i in range(5, 10):
    total = 0
    for j in range(i):
        total += j
    if total % 2 == 0:
        nums.append(i)

print(nums)
```

Calculate the sum of all multiples of 7 from 320 to 945.

Dictionary Analysis

For this problem you will need to use Python to analyze a dictionary. The keys in this dictionary are all strings, and the values are all integers. You will need to compute the sum of all VALUES in the dictionary that are associated with KEYS that contain a ODD number of characters.

For example - if the dictionary is {'ZZZZZ': 5, 'vv': 8, 'vvvvv': 16, 'ff': 4, 'ww': 13} your program would compute the value 21 which was computed by adding 5 + 16 which are associated with keys ('ZZZZZ', 'vvvvv').

The dictionary you are analyzing can be found below (copy this dictionary into your Python editor) - you only need to submit your final answer (an integer)

the_dict = {'HH': 742, 'R': 504, 'VVV': 2690, 's': 44, 'YY': 3988, 'ppp': 4441, 'Y': 4059, 'KKKK': 979, 'BBBBB': 3780, 'mmmm': 2850, 'b': 3028, 'HHHH': 1581, 'PPPPP': 3493, 'jjjj': 4402, 'J': 3466, 'rrrrr': 4226, 'W': 2419, 'TT': 2247, 'RRRR': 3783, 'ZZZ': 3787, 'xxxx': 1831, 'bbbbb': 26, 'cc': 325, 'QQQQQ': 4547, 'c': 2776, 'WWW': 3964, 'kk': 2686, 'CCCC': 3355, 'bbb': 287, 'jjj': 3657, 'C': 1398, 'dddd': 4531, 'pppp': 4227, 'TTT': 4528, 'tt': 1432, 'yyyy': 4654, 'XXX': 619, 'r': 1250, 'ssss': 4556, 'FF': 149, 'ww': 2840, 'HHH': 2615, 'MM': 3741, 'mm': 1286, 'RRR': 237, 'q': 4546, 'VV': 2074, 'mmmmm': 3295, 'sssss': 3694, 'ttttt': 1101, 'yyy': 1074, 'kkkk': 68, 'KKKKK': 2220, 'LL': 842, 'JJ': 2156, 'SSS': 899, 'S': 393, 'CCC': 191, 'ttt': 2973, 'ZZZZZ': 1527, 'VVVV': 3779, 'hhhhh': 967, 'PP': 2759, 'QQQ': 2189, 'qqqqq': 3415, 'YYYY': 1431, 'DDDDD': 2143, 'YYYYY': 4640, 'LLLLL': 1614, 'LLLL': 3260, 'kkk': 1644, 'tttt': 4042, 'YYY': 735, 'K': 3155, 'DD': 3845, 'WWWWW': 4009, 'PPP': 4484, 'rrrr': 1004, 'FFFFF': 3457, 'TTTTT': 1377, 'JJJ': 912, 'D': 1658, 'DDD': 3322, 'BBBB': 2433, 'ccccc': 2651, 'cccc': 1594, 'hh': 607, 'ddddd': 2372, 'XXXXX': 1234, 'M': 2691, 'ZZZZ': 3665, 'PPPP': 1474, 'ss': 3361, 'jj': 2587, 'fff': 1883, 'www': 390, 'vvvvv': 2382, 'wwwww': 1553, 'MMMM': 69, 'wwww': 1906}

Homework

- Midterm next Thursday
- Assignment #6 (due next Thursday)