Introduction to:
Computers & Programming:
Review prior to 2\textsuperscript{nd} Midterm

Adam Meyers
New York University
Summary

• Procedural Matters
• Types of Test Questions and Sample Questions
• Summary of what you need to Know
• Some Tips for Studying
Procedural Matters Regarding the Midterm

• The class will be proctored by 1 or 2 people (depending on class size).
• If possible, there will be at least one empty seat between students
• I will take attendance – please bring your School ID
• The test will be graded on a curve
  – It is difficult for me to figure out how long some of the questions will take, so this may be an important factor
• Each midterm counts for 20% of the final grade
• If you are auditing and want to take the midterm, please let me know in advance and I will try to figure something out.
Type of Questions that Could Be on the Test

• Type 1: Determine What a Program Does
• Type 2: Debug programs
• Type 3: Write programs that answer questions
Example of Type 1: Determine What a Program Does

- The function on the next slide takes a string and outputs the number of time that the word 'bunny' appears in the string. The print statement provides a trace of what is going on in the loop.

- To answer this question, assume the following function call:
  count_the_bunnies('bubunnyxxxbubunny')

- List what the function would print out (this requires that you understand how the loop works)
- List what the function would return.
def count_the_bunnies(input_string):
    count = 0
    bunny = ""
    for character in input_string:
        if (character == 'b'):
            bunny = character
        elif ((character == 'u') and (bunny == 'b')) \
            or ((character == 'n') and ((bunny == 'bu') \
               or (bunny == 'bun'))
               or ((character == 'y') and (bunny == 'bunn'))):
            bunny = bunny + character
        else: bunny = ""
        print('Character:',character,'Bunny:',bunny)
        if bunny == 'bunny':
            count = count+1
            bunny = ""
    return(count)
Type2: Debug a Program

• Find 2 errors in this function and correct them:

```python
def identify_words_that_end_with_ox(string):
    ## this function returns true
    ## if a string ends in 'ox'
    ## and false otherwise
    if ((len(string) >= 2) and 
        (string[-1] == 'o') and 
        (string[len(string)] == 'x')):
        return (true)
    else:
        return(false)
```
Type 3: A Program that Answers a Question

• Write a program that plays a dice game.
• There are 2 dice. You can represent them by choosing a random number from 1 to 6.
  – import random
  – random.random()
  • Picks a random number
  • chooses a number between 0 and 1 which you can modify to make it equivalent to a die
• Each round consists of the computer and the player rolling dice, but the computer wins ties.
• The player starts with 100 points. He gains ten points if he wins and loses ten points if he loses.
• Use a while loop and keep checking if the game ends. The game ends if:
  – The player wins with a point total of 200
  – The player loses with a point total of 0 or less
  – The player quits (and the game says he/she loses for quitting)
What you should know: 1

• How to import modules and reference objects in the modules
  – import turtle
  – my_turtle = turtle.Turtle()

• Basics of turtle graphics
  – You should know how to use them to draw simple shapes

• How to write a function, which is used by another function more than once.

• How to correctly use Boolean operators, if, else and elif
  – How to apply this knowledge to create a decision tree to solve a problem
What you should know 2

• How to use the `input` function
• How to use loops
  – While loops
    while (boolean):
    block-of-text
  – For loops
    for item in sequence:
    block-of-text
  – Endless loops, e.g., clock programs
  – Nested Loops (at most one level of nesting)
What you should know 3

- Sequences:
  - Ranges
  - Strings
  - Indices and Slices

- Working with Loops and Sequences together

- Using variables to count or accumulate results while a loop is running
What you should know 3

- Characters
  - Manipulating characters, e.g., converting characters to their corresponding unicode number, changing the case of a character

- How to make your code clear with:
  - Good variable and function names
  - Comments
  - Function calls
    - Clarify what blocks of code mean
    - Reusable
Some Tips for Studying

• I suspect that the hardest part of these last few weeks is understanding and predicting how loops work.

• Try to do some practice examples. I will try to put a few on line (starting with the ones mentioned above). However, you could probably make up some on your own also.

• It is usually helpful to include a print statement in the loop that gives you an idea how the various variables are updating, e.g., see the following slide.
The two_strings function with helpful print statements

def two_strings():
    string1='abc'
    string2='a'
    while(len(string1)!=len(string2)):
        string1=input('give me a string '
        string2=input('give me another string of equal length')
    output="
    for num in range(len(string1)):
        output=output+string1[num]+string2[num]
        print('string1 character'+str(num)+':',string1[num])
        print('string2 character'+str(num)+':',string2[num])
        print('output so far: ',output)
    return(output)
The Midterm is in 2 Classes

• That means that you have a whole week
  – To prepare as you see fit
  – To ask questions to lab tutors, the email tutor and myself

• Next Class
  – You will have opportunities to discuss problems
  – You have a few days to figure out what those problems are

• Good luck!