

## Caesar Cipher with Files

- High-level description explaining what you're doing at the top of the program
- How to debug
  - Look at the input files
- Common issues
  - Not handling new lines ("\n") in the file
    - Similar to handling spaces
  - Close files as soon as possible
- Testing writing encoded files

Mar 6, 2012

Sprenkle - CSCI111

1

## Review

- What is the keyword we use to create a new function?
- How do we get output from a function?
- What information should a function comment contain?

Mar 6, 2012

Sprenkle - CSCI111

2

## Writing Comments for Functions

- Good style: Each function **must** have a comment
  - Describes functionality at a high-level
  - Include the *precondition*, *postcondition*
  - Describe the **parameters** (their **types**) and the **result** of calling the function (precondition and postcondition may cover this)

Mar 6, 2012

Sprenkle - CSCI111

3

## Writing Comments for Functions

- Include the function's pre- and post-conditions
- **Precondition:** Things that must be true for function to work correctly
  - E.g., num must be even
- **Postcondition:** Things that will be true when function finishes (if precondition is true)
  - E.g., the returned value is the max

Mar 6, 2012

Sprenkle - CSCI111

4

## Function comments

```
def printHeaders():  
    """displays table column headings"""
```

**Good.** Describes function at high level

```
def printHeaders():  
    """defines the printHeader function """
```

Not descriptive.  
Says what *you're* doing, not what **function** does  
Need to tell programmer how to use function

Mar 6, 2012

Sprenkle - CSCI111

5

## TOP-DOWN DESIGN

Mar 6, 2012

Sprenkle - CSCI111

6

## Designing Code

- 1<sup>st</sup> Approach
  - Create functions
  - Call functions
- 2<sup>nd</sup> Approach
  - Write code
  - Refactor code to have functions
  - Call those functions
- Time for 3<sup>rd</sup> approach...
  - Write code, calling functions
  - Write "stub" functions
  - Fill-in functions later

Mar 6, 2012

Sprenkle - CSCI111

7

## Top-Down Design: Alternative Approach to Development

1. Create overview
2. Define functions later

```
def main():
    # get the binary number from the user, as a string
    binNum = input("Please enter a binary number: ")
    isBinary = checkBinary(binNum)
    if not isBinary: # equivalent to isBinary == False
        print(binNum, "is not a binary number.")
        sys.exit()

    decVal = binaryToDecimal(binNum)
    print(binNum, "is", decVal)
```

Mar 6, 2012

Sprenkle - CSCI111

8

## DEAL OR NO DEAL

Mar 6, 2012

Sprenkle - CSCI111

9

## Lab 7: Deal or No Deal Overview

- Have 26 cases with various amounts of money
  - Amounts are known
- Player selects a case (hope has the big jackpot)
- In each round, player opens up cases
  - Reveals amounts that are not in the case they chose
- Banker makes an offer to buy the case
- Player decides if want to take the deal
  - Is the offer more than what is in the case?
  - Make decision based on amounts that haven't been opened yet
- Game ends when only one more case to open (two amounts on board) or player takes the deal.

Mar 6, 2012

Sprenkle - CSCI111

10

## Implementing Deal or No Deal

- Given: partial solution in code
  - main() function, some additional functions are already written
- Your job:
  - Read, understand given code
  - Fill in the functions for a complete solution
- Example of top-down design
  - In main() ... printBoard not yet defined

```
# keep track of how much was in your case
# and mark the case as chosen.
amtInCase = cases[choice]
cases[choice] = CHOSEN
printBoard(caseValues)
```

Mar 6, 2012

11

## Modeling Deal or No Deal

- Cases, numbered 0 to 25
  - Have dollar amounts in them

How can we represent that a case has been opened?

1000000	1000	5	...	750000	value
0	1	2	...	25	case/ position

- Board
  - Which dollar amounts have been chosen, which are still in play

.01	1	5	...	1000000	value
0	1	2	...	25	position

Mar 6, 2012

Sprenkle - CSCI111

12

## Modeling Deal or No Deal

**CHOSEN = -1**  
means case opened:  
Don't display on board,  
Don't allow user to select again

- Cases, numbered 0 to 25

➤ Have dollar amounts in them

1000000	1000	5	...	<b>CHOSEN</b>	value
0	1	2	...	25	case/ position

- Board

➤ Which dollar amounts have been chosen, which are still in play

.01	<b>CHOSEN</b>	5	...	1000000	value
0	1	2	...	25	position

Mar 6, 2012

Sprenkle - CSCI111

13

## Functionality

- Read in values contained in cases from a file
  - What data type should these values be?
- Have user select from remaining cases
  - Make sure choice is valid
- Display remaining cases
  - Print four to a row
- Display remaining amounts on board
  - Left column is smaller amounts

Mar 6, 2012

Sprenkle - CSCI111

14

## How to print remaining cases?

- Cases, numbered 0 to 25

➤ Have dollar amounts in them

1000000	1000	5	...	<b>CHOSEN</b>	value
0	1	2	...	25	case/ position

- Board

➤ Which dollar amounts have been chosen, which are still in play

.01	<b>CHOSEN</b>	1000	...	-1	value
0	1	2	...	25	position

Mar 6, 2012

Sprenkle - CSCI111

15

## Honor System Review

- Person who needs help should *never* look at the code of the person who is helping
- No sharing code
  - No emailing, printing, ...
- Cite the help you're receiving outside of lab
- Pledge your assignments
- Report suspicious behavior

Mar 6, 2012

Sprenkle - CSCI111

16

## Lab 7 Overview

- Focus: program organization
  - Defining and Using Functions
- Deal or No Deal

Mar 6, 2012

Sprenkle - CSCI111

17