Unit 7 Quiz

1. A row-major traversal of a two-dimensional grid visits all of the positions in a: row before moving to the next row
2. The clone method: copies an existing object
3. Instantiation is a process that: creates a new object of a given class
4. The state of an object consists of: the values of all of its attributes
5. In the RGB system, where each color contains three components with 256 possible values each, the number of bits needed to represent each color is: 24
6. The interface of a class is the set of all its: methods
7. The process whereby analog information is converted to digital information is called: sampling
8. The origin (0, 0) in a screen coordinate system is at: the upper-left corner of a window
9. The print function: prints a string representation of an object
10. In a system of 256 unique colors, the number of bits needed to represent each color is 8

Unit 8 Quiz

1. The rows and columns in a grid layout are numbered starting from (0,0)
2. Multi-line text is displayed in a: text area
3. The main window class in a GUI-based program is a subclass of EasyFrame
4. A rectangular subarea with its own grid for organizing widgets is a: panel
5. A window component that supports selecting one option only is the: radio button
6. The attribute used to attach an event-handling method to a button is named:command
7. In contrast to a terminal-based program, a GUI-based program: can allow the user to enter inputs in any order
8. The window component that allows a user to move the text visible beneath a TextArea widget is a: scroll bar
9. The sticky attribute: controls the alignment of a window component in its grid cell
10. GUIs represent color values using: RGB triples of integers

Unit 9 Quiz

1. An instance variable refers to a data value that: is owned by a particular instance of a class and no other
2. A polymorphic method: has a single header but different bodies in different classes.
3. A class variable is used for data that: all instances of a class have in common
4. Class B is a subclass of class A. The _init_ methods in both classes expect no arguments. The call of class A's _init_ method in class B is: A._init_(self)
5. An object's lifetime ends: when it can no longer be referenced anywhere in a program
6. The name used to refer to the current instance of a class within the class definition is: self
7. The purpose of the _init_ method in a class definition is to set the instance variables to initial values
8. A method definition: always must have at least one parameter name, called self
9. The easiest way to save objects to permanent storage is to: pickle them using the pickle function dump
10. The scope of an instance variable is: the entire class in which it is introduced

FINAL

- In Python, functions are treated as first-class data objects. What does this mean? It means that functions can be assigned to variables, passed as arguments to other functions, returned as values, and stored in data structures.
- Which of the following classes are associated with a Turtle object? Screen and Canvas
- What function can you call inside of IDLE to show the resources of the math module, once it has been imported?dir(math)
- What are two common methods by which functions serve as abstraction mechanisms?The elimination of redundant, or repetitious code., The hiding of complicated processes.
- Lists of integers can be built using the range function. T
- Loop statements allow a computer to make choices F
- A difference between functions and methods is that methods cannot return values F
- The code x % y is actually shorthand for the code __mod__(x, y). F
- In programming, what is a prototype? It is a rough draft of a program
- A Caesar cipher uses a plaintext character to compute two or more encrypted characters, and each encrypted character is computed using two or more plaintext characters. F
- Early recording and playback devices for images and sound were all discrete devices F
- What list method should you use in order to remove and return the element at the end of the list named "books"? books.pop()
- In Python, what data type is used to represent real numbers between -10^308 and 10^308 with 16 digits of precision? Float
- All data output to or input from a text file must be strings. T
- Most recursive functions expect no arguments F
- Most object-oriented languages require the programmer to master the following techniques: data encapsulation, inheritance, and abstraction. F
- The while loop is also known as what kind of a loop? Entry-control loop
- The augmented assignment operations have a higher precedence than the standard assignment operation F
- What does a block cipher utilize in order to change plaintext characters into two or more encrypted characters? A mathematical structure known as an invertible matrix
- Each box in a structure chart is labeled with a module name F
- What is the call stack used for in the Python virtual machine? The call stack is an area of reserved memory used to store chunks of memory related to functions
- Objects in the natural world and objects in the world of artifacts can be classified using inheritance hierarchies. T
- A method is always called with a given data value called an object, which is placed before the method name in the call T
- Python is a loosely typed programming language. F
- What os.path module function can you use to convert a path to a pathname appropriate for the current file system? normcase(path)
- Anytime you foresee using a list whose structure will not change, you can, and should, use a tuple instead.T
- As a rule of thumb, when you are defining a new class of objects, you should look around for a class that already supports some of the structure and behavior of such objects, and then subclass that class to provide the service you need. T
- In bottom-up design, you decompose a complex problem into a set of simpler problems and solve these with different functions. F
- A variable associates a name with a value, making it easy to remember and use the value later in a program T
- A Python dictionary is written as a sequence of key/value pairs separated by commas. These pairs are sometimes called indexes. F
- The ticks representing seconds on the analog clock's face represent an attempt to sample moments of time as discrete values, whereas time itself is continuous, or analog. T
- What part of the loop comprises the statements to be executed? The loop body
- Variables receive their initial values and can be reset to new values with an assignment statement. T
- What happens when you make a new class a subclass of another class? The new class inherits the attributes and behaviors defined in the parent class
- Command buttons are created and placed in a window in the same manner as labels. T
- In a GUI-based program, what are labels supposed to do? They are text elements that describe inputs and outputs.
- What is the purpose of a higher-order function?It separates the task of transforming data values from the logic of accumulating the results
- Inheritance allows several different classes to use the same general method names. F

- If the statements in the try clause raise no exceptions, the except clause is skipped and control proceeds to the end of the try-except statement. T
- What type of error is raised when the Python virtual machine runs out of memory resources to manage a process? Stack overflow error
- 4 != 4 evaluates to True. F
- In computer science, what are two names that are used to describe data structures organized by association? (Choose two.) association lists, tables
- In the waterfall development model, what is the most expensive part of software development? The maintenance phase
- Real numbers have infinite precision, which means that the digits in the fractional part can continue forever. T
- What argument can be specified to the two file dialog functions in order to specify the available file types that can be used? filetypes
- A fractal curve is one-dimensional. F
- The not operator expects a single operand and returns its logical negation, True, if it's false, and False if it's true. T
- Like with an infinite loop, an infinite recursion eventually halts execution with an error message. F
- If pop is used with just one argument and this key is absent from the dictionary, Python raises an error. T
- The not operator has a lower precedence than the and and or operators. F
- You are designing a script that may be imported as a module in other scripts. What variable can you use to check whether the script has been imported as a module, or is running as the main script? ___name___
- Smart compilers exist that can optimize some recursive functions by translating them to iterative machine code. T
- Operator overloading is an example of an abstraction mechanism. T
- You are creating a Python script that will make use of user input contact names and their associated phone numbers. What would be ideal for storing and accessing these associated values? Dictionaries
- "A" < "B" evaluates to False. F
- Although a list's elements are always ordered by position, it is possible to impose a natural ordering on them as well. T
- The modulus operator has the highest precedence and is evaluated first. F
- Python programmers typically capitalize their own class names to distinguish them from variable names.T
- What is NOT a Boolean or logical operator used in the Python language? xor
- What is NOT a Boolean or logical operator used in the Python language? F
- The attributes of an object are represented as instance datum. F
- What concept within object-oriented programming involves the restriction of the manipulation of an object's state by external users to a set of method calls? data encapsulation
- When used with strings, the left operand of Python's in operator is a target substring and the right operand is the string to be searched. T
- Which of the following assigns a dictionary list of keys and values to the variable named persons? persons = {"Sam":"CEO", "Julie":"President"}
- In order to allow a program to respond to a button click, the programmer must set the button's "response" attribute. F

Notes on Final website:

Must function!
Change:
- Name of pages
- Text
- Images

Should include the following pages:
- Home
- Skills
- Biography
- Job experience
- References
- Services
- Personal interest
- Contact

Add contact form

9.1

```
"""
File: student.py
Resources to manage a student's name and test scores.
"""

class Student(object):
    """Represents a student."""

    def __init__(self, name, number):
        """All scores are initially 0."""
        self.name = name
        self.scores = []
        for count in range(number):
            self.scores.append(0)

    def getName(self):
        """Returns the student's name."""
        return self.name

    def setScore(self, i, score):
        """Resets the ith score, counting from 1."""
        self.scores[i - 1] = score

    def getScore(self, i):
        """Returns the ith score, counting from 1."""
        return self.scores[i - 1]

    def getAverage(self):
        """Returns the average score."""
        return sum(self.scores) / len(self._scores)

    def getHighScore(self):
        """Returns the highest score."""
        return max(self.scores)

    def __str__(self):
        """Returns the string representation of the student."""
        return "Name: " + self.name  + "\nScores: " + \
               " ".join(map(str, self.scores))

    # Write method definitions here
    def __eq__(self, other):
            """Tests for equality."""
            if self is other:
                return True
            elif type(self) != type(other):
                return False
            else:
                return self.name == other.name
    def __lt__(self, other):
            """Returns self < other, with respect
            to names."""
            return self.name < other.name

    def __ge__(self, other):
        """Returns self >= other, with respect
        to names."""
        return self.name >= other.name

def main():
    """A simple test."""
    student = Student("Ken", 5)
    print(student)
    for i in range(1, 6):
        student.setScore(i, 100)
    print(student)

if __name__ == "__main__":
    main()
```

9.2

```
import random
```

```
class Student(object):
    """Represents a student."""

    def __init__(self, name, number):
        """All scores are initially 0."""
        self.name = name
        self.scores = []
        for count in range(number):
            self.scores.append(0)

    def getName(self):
        """Returns the student's name."""
        return self.name

    def setScore(self, i, score):
        """Resets the ith score, counting from 1."""
        self.scores[i - 1] = score

    def getScore(self, i):
        """Returns the ith score, counting from 1."""
        return self.scores[i - 1]

    def getAverage(self):
        """Returns the average score."""
        return sum(self.scores) / len(self._scores)

    def getHighScore(self):
        """Returns the highest score."""
        return max(self.scores)

    def __str__(self):
        """Returns the string representation of the student."""
        return "Name: " + self.name  + "\nScores: " + \
               " ".join(map(str, self.scores))

    def __lt__(self, other):
        """Returns self < other, with respect
        to names."""
        return self.name < other.name

    def __ge__(self, other):
        """Returns self >= other, with respect
        to names."""
        return self.name >= other.name

    def __eq__(self, other):
        """Tests for equality."""
        if self is other:
            return True
        elif type(self) != type(other):
            return False
        else:
```

```
        return self.name == other.name

def main():
    """Tests sorting."""
    # Create the list and put 5 students into it
    lyst = []
    for count in reversed(range(5)):
        s = Student("Name" + str(count + 1), 10)
        lyst.append(s)
    random.shuffle(lyst)
    print("Unsorted list of students")
    for s in lyst:
        print(s)
    lyst.sort()
    print("\nSorted list of students:")
    for s in lyst:
        print(s)
    # Complete the definition of the main function

if __name__ == "__main__":
    main()
```

9.3

*** I think changes were only made for bank.py but could be wrong***

```
"""
File: bank.py
This module defines the Bank class.
"""
import pickle
import random
from savingsaccount import SavingsAccount

class Bank:
    """This class represents a bank as a collection of savings accounts.
    An optional file name is also associated
    with the bank, to allow transfer of accounts to and
    from permanent file storage."""

    """The state of the bank is a dictionary of accounts and
    a file name.  If the file name is None, a file name
    for the bank has not yet been established."""

    def __init__(self, fileName = None):
        """Creates a new dictionary to hold the accounts.
        If a file name is provided, loads the accounts from
        a file of pickled accounts."""
        self.accounts = {}
        self.fileName = fileName
        if fileName != None:
            fileObj = open(fileName, 'rb')
            while True:
                try:
                    account = pickle.load(fileObj)
                    self.add(account)
                except Exception:
                    fileObj.close()
                    break

    def __str__(self):
        """Returns the string representation of the bank."""

        # using the sorted method sorting the dictionary
        # by values's name
        return "\n".join([str(v) for (k, v) in
                         sorted(self.accounts.items(),
                              key=lambda cv: cv[1].getName())])
    def makeKey(self, name, pin):
        """Returns a key for the account."""
        return name + "/" + pin

    def add(self, account):
        """Adds the account to the bank."""
        key = self.makeKey(account.getName(), account.getPin())
        self.accounts[key] = account

    def remove(self, name, pin):
        """Removes the account from the bank and
        and returns it, or None if the account does
        not exist."""
        key = self.makeKey(name, pin)
        return self.accounts.pop(key, None)

    def get(self, name, pin):
        """Returns the account from the bank,
        or returns None if the account does
        not exist."""
        key = self.makeKey(name, pin)
        return self.accounts.get(key, None)

    def computeInterest(self):
        """Computes and returns the interest on
        all accounts."""
        total = 0
        for account in self._accounts.values():
            total += account.computeInterest()
        return total

    def getKeys(self):
        """Returns a sorted list of keys."""
        # Exercise
        return []

    def save(self, fileName = None):
        """Saves pickled accounts to a file.  The parameter
        allows the user to change file names."""
        if fileName != None:
            self.fileName = fileName
        elif self.fileName == None:
            return
        fileObj = open(self.fileName, 'wb')
        for account in self.accounts.values():
            pickle.dump(account, fileObj)
        fileObj.close()

# Functions for testing

def createBank(numAccounts = 1):
    """Returns a new bank with the given number of
    accounts."""
    names = ("Brandon", "Molly", "Elena", "Mark", "Tricia",
             "Ken", "Jill", "Jack")
    bank = Bank()
    upperPin = numAccounts + 1000
    for pinNumber in range(1000, upperPin):
        name = random.choice(names)
        balance = float(random.randint(100, 1000))
        bank.add(SavingsAccount(name, str(pinNumber), balance))
    return bank

def testAccount():
    """Test function for savings account."""
    account = SavingsAccount("Ken", "1000", 500.00)
    print(account)
    print(account.deposit(100))
    print("Expect 600:", account.getBalance())
    print(account.deposit(-50))
    print("Expect 600:", account.getBalance())
    print(account.withdraw(100))
    print("Expect 500:", account.getBalance())
    print(account.withdraw(-50))
    print("Expect 500:", account.getBalance())
    print(account.withdraw(100000))
    print("Expect 500:", account.getBalance())

def main(number = 10, fileName = None):
    """Creates and prints a bank, either from
    the optional file name argument or from the optional
    number."""
    testAccount()

    print("\n----------Account details------------\n")
    if fileName:
        bank = Bank(fileName)
    else:
        bank = createBank(number)
    print(bank)

if __name__ == "__main__":
    main()

----------    -----------------    ---------------    ---------------------
"""
File: savingsaccount.py
This module defines the SavingsAccount class.
"""

class SavingsAccount:
    """This class represents a savings account
    with the owner's name, PIN, and balance."""

    RATE = 0.02    # Single rate for all accounts

    def __init__(self, name, pin, balance = 0.0):
        self.name = name
        self.pin = pin
        self.balance = balance
```

```
def __str__(self):
    """Returns the string rep."""
    result =  'Name:    ' + self.name + '\n'
    result += 'PIN:     ' + self.pin + '\n'
    result += 'Balance: ' + str(self.balance)
    return result

def getBalance(self):
    """Returns the current balance."""
    return self.balance

def getName(self):
    """Returns the current name."""
    return self.name

def getPin(self):
    """Returns the current pin."""
    return self.pin

def deposit(self, amount):
    """If the amount is valid, adds it
    to the balance and returns None;
    otherwise, returns an error message."""
    self.balance += amount
    return None

def withdraw(self, amount):
    """If the amount is valid, sunstract it
    from the balance and returns None;
    otherwise, returns an error message."""
    if amount < 0:
        return "Amount must be >= 0"
    elif self.balance < amount:
        return "Insufficient funds"
    else:
        self.balance -= amount
        return None

def computeInterest(self):
    """Computes, deposits, and returns the interest."""
    interest = self.balance * SavingsAccount.RATE
    self.deposit(interest)
    return interest
```

9.5

in Doctory.py, change the greeting and farewell messages if you want

```
"""
File: doctor.py
"""
import random

class Doctor(object):

    history = []

    hedges = ("Please tell me more.",
              "Many of my patients tell me the same thing.",
              "Please coninue.")

    qualifiers = ("Why do you say that ",
                  "You seem to think that ",
                  "Can you explain why ")

    replacements = {"I": "you", "me": "you", "my": "your",
                    "we": "you", "us": "you", "mine": "yours",
                    "you": "I", "your": "my", "yours": "mine"}

    def __init__(self):
        pass

    def reply(self, sentence):
        """Implements three different reply strategies."""
        probability = random.randint(1, 5)
        if probability in (1, 2):
            # Just hedge
            answer = random.choice(Doctor.hedges)
        elif probability == 3 and len(Doctor.history) > 3:
            # Go back to an earlier topic
            answer = "Earlier you said that " + \
                self.changePerson(random.choice(Doctor.history))
        else:
            # Transform the current input
            answer = random.choice(Doctor.qualifiers) + \
                self.changePerson(sentence)

        Doctor.history.append(sentence)
        return answer

    def changePerson(self, sentence):

        words = sentence.split()
        replyWords = []
        for word in words:
            replyWords.append(Doctor.replacements.get(word, word))
        return " ".join(replyWords)
    def greeting(self):
        return "Hi, Let's get started"

    def farewell(self):
        return "Time's up, Let's pick up here next week"
```

***create a new file: doctor main ***

```
def main():
    d = Doctor()
    print(d.greeting())
    while True:
        sentence = input("\n>> ")
        if sentence.upper() == "QUIT":
            print(d.farewell())
            break
        print(d.reply(sentence))

if __name__ == "__main__":
    main()
```

```
$ python doctormain.py
Good morning, I hope you are well today

>> have checkup
Please tell me more.

>> am not sure
Can you explain why am not sure

>> dont know
Please tell me more.

>> what
Why do you say that what

>> QUIT
Have a nice day!
```

Please do let me know if u have any concern...

9.6

```
"""
craps.py
Project 6
This module studies and plays the game of craps.
Refactors code from case study so that the user
can have the Player object roll the dice and view
the result.
"""

from die import Die

class Player(object):

    def __init__(self):
        """Has a pair of dice and an empty rolls list."""
        self.die1 = Die()
        self.die2 = Die()
        self.roll = ""
        self.rollsCount = 0
        self.atStartup = True
        self.winner = self.loser = False

    def __str__(self):
        """Returns a string representation of the last roll."""
        return self.roll

    def getNumberOfRolls(self):
        """Returns the number of the rolls."""
        return self.rollsCount

    def rollDice(self):
        """Rolls the dice once. Updates the roll, the won and
        lost outcomes, and returns a tuple of the values
        of the dice."""
        self.rollsCount += 1
        self.die1.roll()
        self.die2.roll()
        (v1, v2) = (self.die1.getValue(),
                    self.die2.getValue())
        self.roll = str((v1, v2)) + " total = " + str(v1 + v2)
        # Game logic for one roll of the dice.
        if self.atStartup:
            self.initialSum = v1 + v2
            self.atStartup = False
            if self.initialSum in (2, 3, 12):
                self.loser = True
            elif self.initialSum in (7, 11):
                self.winner = True
        else:
            laterSum = v1 + v2
            if laterSum == 7:
                self.loser = True
            elif laterSum == self.initialSum:
                self.winner = True
        return (v1, v2)

    # Note: both isWinner() and isLoser() can be False,
    # if the game is not finished.

    def isWinner(self):
        """Returns True if player has won."""
        return self.winner

    def isLoser(self):
        """Returns True if player has lost."""
        return self.loser

    def play(self):
        """Plays a game, counts the rolls for that game,
        and returns True for a win and False for a loss."""
        while not self.isWinner() and not self.isLoser():
            self.rollDice()
        return self.isWinner()

def playOneGame():
    """Plays a single game and prints the results after
    each roll."""
    player = Player()
    while not player.isWinner() and not player.isLoser():
        player.rollDice()
        print(player)
    if player.isWinner():
        print("You win!")
    else:
        print("You lose!")

def playManyGames(number):
    """Plays a number of games and prints statistics."""
    wins = 0
    losses = 0
    winRolls = 0
    lossRolls = 0
    for count in range(number):
        player = Player()
        hasWon = player.play()
        rolls = player.getNumberOfRolls()
        if hasWon:
            wins += 1
            winRolls += rolls
        else:
            losses += 1
            lossRolls += rolls
    print("The total number of wins is", wins)
    print("The total number of losses is", losses)
    print("The average number of rolls per win is %0.2f" % \
          (winRolls / wins))
    print("The average number of rolls per loss is %0.2f" % \
          (lossRolls / losses))
    print("The winning percentage is %0.3f" % (wins / number))

def main():
    """Plays one game and then a number of games and prints statistics."""
    playOneGame()
    number = int(input("Enter the number of games: "))
    playManyGames(number)

if __name__ == "__main__":
    main()
```

9.7

crapsgui.py

```
from breezypythongui import EasyFrame
from tkinter import PhotoImage
from craps import Player
class CrapsGUI(EasyFrame):
    def __init__(self):
        EasyFrame.__init__(self,title="Craps Game")
        self.setSize(200,200)
        self.player=Player()
        self.v1=1
        self.v2=2
        self.dieLabel1=self.addLabel("",row=0,column=0,sticky="NSEW")
        self.dieLabel2=self.addLabel("",row=0,column=1,sticky="NSEW")
        self.stateArea=self.addTextArea("",row=1,column=0,columnspan=2,width=15,height=5)
        self.rollButton=self.addButton(row=2,column=0,text="Roll",command=self.nextRoll)
        self.addButton(row=2,column=1,text="New Game",command=self.newGame)
        self.refreshImages()
    def nextRoll(self):
        (self.v1,self.v2)=self.player.rollDice()
        total=self.v1+self.v2
        self.stateArea.appendText("Total = "+str(total)+"\n")
        self.refreshImages()
        if self.player.isWinner():
            self.stateArea.appendText("You won!")
            self.rollButton["state"]="disabled"
        elif self.player.isLoser():
            self.stateArea.appendText("You lose!")
            self.rollButton["state"]="disabled"
    def newGame(self):
        self.player=Player()
        self.v1=1
        self.v2=1
        self.stateArea.setText("")
        self.refreshImages()
        self.rollButton["state"]="normal"
    def refreshImages(self):
        fileName1="DICE/"+str(self.v1)+".gif"
        fileName2="DICE/"+str(self.v2)+".gif"
        self.image1=PhotoImage(file=fileName1)
        self.dieLabel1["image"]=self.image1
        self.image2=PhotoImage(file=fileName2)
        self.dieLabel2["image"]=self.image2
def main():
    CrapsGUI().mainloop()
if __name__=="main":
    main()
```

9.8

```
"""
File: cards.py

Module for playing cards, with classes Card and Deck
"""
import random

class Card(object):
    """ A card object with a suit and rank."""

    RANKS = (1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13)

    SUITS = ('Spades', 'Diamonds', 'Hearts', 'Clubs')

    def __init__(self, rank, suit):
        """Creates a card with the given rank and suit."""
        self.rank = rank
        self.suit = suit
        self.faceup = False

    def turn(self):
        self.faceup = not self.faceup

    def __str__(self):
        """Returns the string representation of a card."""
        if self.rank == 1:
            rank = 'Ace'
        elif self.rank == 11:
            rank = 'Jack'
        elif self.rank == 12:
            rank = 'Queen'
        elif self.rank == 13:
            rank = 'King'
        else:
            rank = self.rank
        return str(rank) + ' of ' + self.suit

import random

class Deck(object):
    """ A deck containing 52 cards."""

    def __init__(self):
        """Creates a full deck of cards."""
        self.cards = []
        for suit in Card.SUITS:
            for rank in Card.RANKS:
                c = Card(rank, suit)
                self.cards.append(c)
    def shuffle(self):
        """Shuffles the cards."""
        random.shuffle(self.cards)

    def deal(self):
        """Removes and returns the top card or None
        if the deck is empty."""
        if len(self) == 0:
           return None
        else:
           return self.cards.pop(0)

    def __len__(self):
       """Returns the number of cards left in the deck."""
       return len(self.cards)

    def __str__(self):
        """Returns the string representation of a deck."""
        result = ''
        for c in self.cards:
            result = self.result + str(c) + '\n'
        return result

def main():
    """A simple test."""
    deck = Deck()
    print("A new deck:")
    while len(deck) > 0:
        print(deck.deal())
    deck = Deck()
    deck.shuffle()
    print("A deck shuffled once:")
    while len(deck) > 0:
        print(deck.deal())

if __name__ == "__main__":
    main()
```

Mid-Term 2 Practice

- Due Nov 25

1. What is the "self" parameter utilized for in a class definition block?It is used within the class and method definitions to call other methods on the same object, or to access that object's variables or data
2. A GUI-based program displays a window that contains various components, also known by what specific term? Widgets
3. An abstraction hides detail and thus allows a person to view many things as just one thing. T
4. A Python dictionary is written as a sequence of key/value pairs separated by commas. These pairs are sometimes called indexes. F
5. What are the two arguments expected by the get method for a dictionary object? (Choose two.) A possible key, A default value
6. A function can be defined in a Python shell, but it is more convenient to define it in an IDLE window, where it can be saved to a file: T
7. What is the name for a diagram that shows the relationships among a program's functions and the passage of data between them? structure chart
8. When using the EasyFrame class to create a window, what method specifically is used to set up any window components to display in the window?__init__
9. The rectangular display area on a computer screen is made up of colored dots called picture elements or pixels. T
10. Where can the required arguments for a function be found? In the function header.
11. Recursive functions are frequently used to design algorithms for computing values that have a recursive definition.T
12. The == operator returns True if the variables are aliases for the same object. Unfortunately, == returns False if the contents of two different objects are the same. F
13. The coordinate system for turtle graphics is the standard Cartesian system, with the origin (0, 0) at the bottom-left corner of a window. F
14. GUI-based programs are almost always object-oriented. T
15. sticky attribute are "T, D, L, R". False
16. What term describes a dictionary of functions keyed by command names? Jump table
17. The images module is a standard, open-source Python tool: F
18. A recursive function must contain at least one repetition statement F
19. Which of the following image formats is considered to be the most popular? (Choose two.) JPEG, GIF
20. A list is a sequence of data values called indexes: F
21. In the RGB system, the value #00ff00 represents the color red: F
22. What statement accurately describes what a mutator is in the Python language?A mutator is a method that is devoted entirely to the modification of the internal state of an object
23. If you use the range method with a single parameter of 50, what will be the range of integers included in the returned list? 0 through 49
24. The index of the first item in a list is 1. F
25. The values that can be used within the sticky attribute are "T, D, L, R". F

Mid-Term 2

26. A list is a sequence of data values known by either of what two terms? (Choose two.) elements, items
27. What RGB value should you use to produce a blue color? 0,0,255
28. An Identity function usually tests its argument for the presence or absence of some property: F
29. An entry field is a box in which the user can position the mouse cursor and enter a number or a single line of text. T
30. The index of the last item in a list is the length of the list. F
31. In turtle graphics, the turtle is initially facing east. F
32. A button has a state attribute, which can be set to "enabled" or "disabled": F
33. What are the two functions available in the tkinter.filedialog module used to support file access in a GUI program? asksaveasfilename, askopenfilename
34. What statement accurately describes the difference between a list and a tuple? A tuple's contents cannot be changed.
35. One of the requirements for a breezypythongui-based Python script you're writing is that you be able to use a slider bar for selecting a value from a range of values. What type of window component do you want? Scale
36. The size of a pixel is determined by the size and resolution of the display: T
37. GUI-based programs use windows that contain various components, also called window objects or gadgets. F
38. The breezypythongui module's IntegerField component is a subclass of the Entry class. T
39. The list method ascending mutates a list by arranging its elements in ascending order. F
40. In a dictionary, a -> separates a key and its value. F
41. What is instantiation in Python?It is the process of creating a class-based object
42. What statement accurately describes the difference between a list and a tuple? A tuple's contents cannot be changed
43. The definition of the main function and the other function definitions can appear in no particular order in the script, as long as main is called at the very end of the script. T
44. RGB stands for red, green, and black. F
45. Given a Turtle object of t, what method can you use to erase all of the turtle's drawings, without changing the state? t.erase()

Ch 8 Quiz

46. When using the EasyFrame class to create a window, what method specifically is used to set up any window components to display in the window? __init__
47. GUI-based programs can be configured to respond to various keyboard events and mouse events. T
48. A GUI-based program displays a window that contains various components, also known by what specific term? Widgets
49. By default, a window has an empty title string and is resizable. T
50. The addButton method returns an object of type tkinter.Button. T
51. Constructors are used to create new instances of a class. T
52. A method header looks very different from a function header. F
53. GUI-based programs utilize file dialogs in order to allow the user to browse a computer's file system. T
54. A black-and-white photograph contains more than just the black and white colors, but various shades of gray known by what term? Grayscale
55. Lists of strings can be displayed in list boxes: T
56. What is NOT one of the more important attributes of a window? The terminal prompt
57. The terminal prompt: t.erase()
58. Each item in a list has a unique index that specifies its position: T
59. The main function usually expects no arguments and returns no value; T
60. In the RGB system, each color component can range from 0 through 127: F
61. Command buttons can display text but not images. F
62. What is the name of the standard GUI module for Python? tkinter
63. The drawing of simple, two-dimensional shapes, such as rectangles, triangles, pentagons, and circles, is known as what type of graphics? Vector graphics

Unit 8 Reviewing the Basics Quiz

64. The sticky attribute **controls the alignment of a window component in its grid cell**
65. Multi-line text is displayed in a **text area**
66. A window component that supports selecting one option only is the **radio button**
67. The main window class in a GUI-based program is a subclass of **EasyFrame**
68. A rectangular subarea with its own grid for organizing widgets is a **panel**
69. The rows and columns in a grid layout are numbered starting from **(0, 0)**
70. GUIs represent color values using **RGB triples of integers**
71. In contrast to a terminal-based program, a GUI-based program **can allow the user to enter inputs in any order**
72. The window component that allows a user to move the text visible beneath a TextArea widget is a **scroll bar**
73. The attribute used to attach an event-handling method to a button is named **command**

8.7

```
from breezypythongui import EasyFrame

class TidbitGUI(EasyFrame):
   """Demonstrates a multiline text area."""

   def __init__(self):
      """Sets up the window and widgets."""
      EasyFrame.__init__(self, "Loan Amortization")
      self.addLabel(text = "Initial amount", row = 0, column = 0)

      self.addLabel(text = "Interest rate in %", row = 1, column = 0)
      self.priceField = self.addFloatField(value = 0.0, row = 0, column = 1)

      self.rateField = self.addIntegerField(value = 0, row = 1, column = 1)

      self.outputArea = self.addTextArea("", row = 3, column = 0,
                                          columnspan = 6,
                                          width = 95, height = 25)

      self.button = self.addButton(text = "Compute", row = 2, column = 0,
                                   columnspan = 2,
                                   command = self.compute)
   # Event handling method.
   def compute(self):
      """Computes the investment schedule based on the inputs
      and outputs the schedule."""
      # Obtain and validate the inputs
      purchasePrice = self.priceField.getNumber()
      rate = self.rateField.getNumber() / 100
      if purchasePrice == 0 or rate == 0:
         return

      loanAmount = purchasePrice
      payment = (loanAmount - (.10 * purchasePrice) ) * 0.05
      numMonths = int(loanAmount//payment)

      # Set the header for the table
      result ="%8s%19s%18s%19s%10s%17s\n" % \
         ("Month", "Starting Balance", "Interest to Pay", "Principal to Pay", "Payment", "Ending Balance")

      # Compute and append the results for each yeafor month in range(1, numMonths+1):
        startingPrice = purchasePrice
        for month in range(1, numMonths+1):
            interesttoPay = loanAmount * rate /12
            endingBalance = loanAmount - payment
            principaltoPay = payment - interesttoPay
             # Display the results
            result += "%8d%19.2f%18.2f%19.2f%10.2f%17.2f\n" % \
                             (month, loanAmount,interesttoPay, principaltoPay, payment, endingBalance)
            loanAmount = endingBalance
            month += 1


        # Output the result while preserving read-only status
        self.outputArea["state"] = "normal"
        self.outputArea.setText(result)
        self.outputArea["state"] = "disabled"

def main():
    """Instantiate and pop up the window."""
    TidbitGUI().mainloop()

if __name__ == "__main__":
    main()
```

8.6

```
"""
File: taxformwithgui.py
Project 8.6
A GUI-based tax calculator.

Computes and prints the total tax, given the income and
number of dependents (inputs), and a standard deduction of
$10,000, an exemption amount of $3,000, and tax rates of
20% for Single
15% for Married
10% for Divorced
"""

from breezypythongui import EasyFrame

class TaxCalculator(EasyFrame):
    """Application window for the tax calculator."""

    def __init__(self):
        """Sets up the window and the widgets."""
        EasyFrame.__init__(self, title = "Tax Calculator")

        # Label and field for the income
        self.addLabel(text = "Gross Income", row = 0, column = 0)
        # (self.incomeField)
        self.incomeField = self.addFloatField(value = 0.0, row = 0, column =1)
        # Label and field for the number of dependents
        self.addLabel(text="Dependents", row =1, column = 0)
        # (self.depField)
        self.depField = self.addIntegerField(value = 0, row = 1, column = 1)
        # Radio buttons for filing status
        self.statusGroup = self.addRadiobuttonGroup(row = 0, column=2, rowspan =3)

        # Button group (self.statusGroup)
        #
        # Option for single (self.single)
        self.single = self.statusGroup.addRadiobutton(text = "Single")
        # Option for married (self.married)
        self.married = self.statusGroup.addRadiobutton(text = "Married")
        # Option for divorced (self.divorced)
        self.divorced = self.statusGroup.addRadiobutton(text = "Divorced")
        # The compute button
        self.addButton(text = "Compute", row=2, column =0, columnspan = 2, command = self.computeTax)
        # Label and field for the tax
        self.addLabel(text = "Total Tax", row = 3, column = 0)
        # (self.taxField)
        self.taxField = self.addFloatField(value=0.0, row = 3, column = 1, precision = 2, state = "readonly")
    # The event handler method for the button
    def computeTax(self):
        """Obtains the data from the input field and uses
        them to compute the tax, which is sent to the
        output field (taxField)."""
        rate = 0
        status = self.statusGroup.getSelectedButton()
        if status == self.single:
            rate = .20
        elif status == self.married:
            rate = .15
        elif status == self.divorced:
            rate = .10
        income = self.incomeField.getNumber()
        numDependents = self.depField.getNumber()
        standardDeduction = 10000.0
        exemptionAmount = 3000.0
        tax = (income - numDependents * exemptionAmount - standardDeduction) * rate
        self.taxField.setNumber(max(tax, 0))



def main():
    TaxCalculator().mainloop()

if __name__ == "__main__":
    main()
```

8.5

```
"""
File: guesswithgui.py
Project 8.5
The computer guesses a number and the user provides the hints.
"""

import random
from breezypythongui import EasyFrame

class GuessingGame(EasyFrame):
    """Plays a guessing game with the user."""

    def __init__(self):
        """Sets up the window,widgets, and data."""
        EasyFrame.__init__(self, title = "Guessing Game")
        self.lowerBound = 1
        self.upperBound = 100
        self.count = 0
        self.myNumber = (self.lowerBound + self.upperBound) // 2
        guess = "Is the number " + str(self.myNumber) + "?"
        self.myLabel = self.addLabel(text = guess,
                                      row = 0, column = 0,
                                      sticky = "NSEW",
                                      columnspan = 4)
        self.small = self.addButton(text = "Too small", row = 1,
                                    column = 0, command = self.goLarge)
        self.large = self.addButton(text = "Too large", row = 1,
                                      column = 1,
                                      command = self.goSmall)
        self.correct = self.addButton(text = "Correct", row = 1,
                                      column = 2,
                                      command = self.goCorrect)
        self.newButton = self.addButton(text = "New game", row = 1,
                                      column = 3,
                                      command = self.newGame)

    def goLarge(self):
        """Guess was too small, so move guess to the right of the number."""
        self.count += 1
        self.lowerBound = self.myNumber +1
        self.myNumber = (self.lowerBound + self.upperBound)//2
        guess = "Is the number"+ str(self.myNumber)+ "?"
        self.myLabel["text"] = guess
    def goSmall(self):
        """Guess was too large, so move guess to the left of the number."""
        self.count += 1
        self.upperBound = self.myNumber -1
        self.myNumber = (self.lowerBound + self.upperBound)//2
        guess = "Is the number" + str(self.myNumber)+ "?"
        self.myLabel["text"] = guess

    def goCorrect(self):
        """Guess was too correct, so announce and wait."""
        self.count += 1
        guess = "I've got it in" + str(self.count)+ "tries!"
        self.myLabel["text"] = guess
        self.small["state"] = "disabled"
        self.large["state"] = "disabled"
        self.correct["state"]= "disabled"

    def newGame(self):
        """Resets the GUI to its original state."""
        self.lowerBound = 1
        self.upperBound = 100
        self.count = 0
        self.myNumber = (self.lowerBound + self.upperBound)//2
        guess = "Is the number" + str(self.myNumber) + "?"
        self.myLabel["text"] = guess
        self.small["state"] = "normal"
        self.large["state"]= "normal"
        self.correct["state"] = "normal"

def main():
    """Instantiate and pop up the window."""
    GuessingGame().mainloop()

if __name__ == "__main__":
    main()
```

8.4

```
"""
File: temperatureconverter.py
Project 8.4
Temperature conversion between Fahrenheit and Celsius.
Illustrates the use of numeric data fields.
Responds to a return key event in the entry fields.
"""
from breezypythongui import EasyFrame

class TemperatureConverter(EasyFrame):
    """A termperature conversion program."""

    def __init__(self):
        """Sets up the window and widgets."""
        EasyFrame.__init__(self, title = "Temperature Converter")

        # Label and field for Celsius
        self.addLabel(text = "Celsius",
                      row = 0, column = 0)
        self.celsiusField = self.addFloatField(value = 0.0,
                                               row = 1,
                                               column = 0,
                                               precision = 2)

        # Label and field for Fahrenheit
        self.addLabel(text = "Fahrenheit",
                      row = 0, column = 1)
        self.fahrField = self.addFloatField(value = 32.0,
                                            row = 1,
                                            column = 1,
                                            precision = 2)

        # Celsius to Fahrenheit button
        self.addButton(text = ">>>>",
                       row = 2, column = 0,
                       command = self.computeFahr)

        # Fahrenheit to Celsius button
        self.addButton(text = "<<<<",
                       row = 2, column = 1,
                       command = self.computeCelsius)
        # Enalbe event handlers for return key in fields
        self.celsiusField.bind("<Return>", lambda event: self.computeFahr())
        self.fahrField.bind("<Return>", lambda event:self.computeCelsius())

    # The controller methods
    def computeFahr(self):
        """Inputs the Celsius degrees
        and outputs the Fahrenheit degrees."""
        degrees = self.celsiusField.getNumber()
        degrees = degrees * 9 / 5 + 32
        self.fahrField.setNumber(degrees)
    def computeCelsius(self):
        """Inputs the Fahrenheit degrees
        and outputs the Celsius degrees."""
        degrees = self.fahrField.getNumber()
        degrees = (degrees - 32) * 5 / 9
        self.celsiusField.setNumber(degrees)

def main():
    """Instantiate and pop up the window."""
    TemperatureConverter().mainloop()

if __name__ == "__main__":
    main()
```

8.3

```
'''
File: temperatureconverter.py
Project 8.3
Temperature conversion between Fahrenheit and Celsius.
Illustrates the use of numeric data fields.

'''
from breezypythongui import EasyFrame

class TemperatureConverter(EasyFrame):
    """A termperature conversion program."""

    def __init__(self):
        """Sets up the window and widgets."""
        EasyFrame.__init__(self, title = "Temperature Converter")

        # Label and field for Celsius
        self.addLabel(text = "Celsius",
                      row = 0, column = 0)
        self.celsiusField = self.addFloatField(value = 0.0,
                                               row = 1,
                                               column = 0,
                                               precision = 2)

        # Label and field for Fahrenheit
        self.addLabel(text = "Fahrenheit",
                      row = 0, column = 1)
        self.fahrField = self.addFloatField(value = 32.0,
                                            row = 1,
                                            column = 1,
                                            precision = 2)

        # Celsius to Fahrenheit button
        self.addButton(text = ">>>>",
                       row = 2, column = 0,
                       command = self.computeFahr)

        # Fahrenheit to Celsius button
        self.addButton(text = "<<<<",
                       row = 2, column = 1,
                       command = self.computeCelsius)

    # The controller methods
    def computeFahr(self):
        """Inputs the Celsius degrees
        and outputs the Fahrenheit degrees."""
        degrees = self.celsiusField.getNumber()
        degrees = degrees * 9 / 5 + 32
        self.fahrField.setNumber(degrees)

    def computeCelsius(self):
        """Inputs the Fahrenheit degrees
        and outputs the Celsius degrees."""
        degrees = self.fahrField.getNumber()
        degrees = (degrees - 32) * 5 / 9
        self.celsiusField.setNumber(degrees)

def main():
    """Instantiate and pop up the window."""
    TemperatureConverter().mainloop()

if __name__ == "__main__":
    main(
```

--8.2--

```
"""
File: bouncywithgui.py
```

8.2

```
Determines the distance traveled by a bouncing ball.

Inputs: Initial height, bounciness index, and number of bounces
"""

from breezypythongui import EasyFrame

def computeDistance(height, index, bounces):
    """Computes the total distance traveled by the ball,
    given an initial height, bounciness index, and
    number of bounces."""
    pass

class BouncyGUI(EasyFrame):

    def __init__(self):
        """Set up the window and widgets."""
        EasyFrame.__init__(self,  title = "Bouncy")
        # Define the following fields:
        # * self.heightField (number entry)
        self.addLabel(text = "Initial Height",row = 0, column = 0)
        self.heightField = self.addIntegerField(value = 0,row = 0,column = 1, width = 10)
        # * self.indexField (number entry)
        self.addLabel(text = "Index",row = 1, column = 0)
        self.indexField = self.addFloatField(value = 0,row = 1,column = 1, width = 10)
        # * self.bouncesField (number entry)
        self.addLabel(text = "Number of Bounces",row = 2, column = 0)
        self.bouncesField = self.addIntegerField(value = 0,row = 2,column = 1, width = 10)
        # * self.distanceField (result result)
        self.compuuteButton = self.addButton(text = "Compute",row = 3, column =0,columnspan =2,command =
self.computeDistance)
        self.addLabel(text = "Distance",row = 4, column = 0)
        self.distanceField = self.addFloatField(value = 0,row = 4,column = 1, width = 10)

    def computeDistance(self):
        """
        Event handler for the Compute button and set the
        distanceField.
        """
        height=self.heightField.getNumber()
        index=self.indexField.getNumber()
        bounces=self.bouncesField.getNumber()
        distance=0
        for eachPass in range(bounces):
            distance+=height
            height *=index
            distance+=height
        self.distanceField.setNumber(distance)




def main():
    """Instantiate and pop up the window."""
    BouncyGUI().mainloop()


if __name__ == "__main__":
    main()



---------------------------------------8.2------------------------------------------------


---------------------------------------8.1------------------------------------------------


"""
```

7.5

posterize.py
File: taxformwithgui.py

```
from images import Image

def posterize(image, color=(0,0,0)):
    """Converts the color image to two coor, one of which is white and     the other which either a default of black or a user specified RGB     value."""
    whitePixel = (255, 255, 255)
    for y in range(image.getHeight()):
        for x in range(image.getWidth()):
            (r, g, b) = image.getPixel(x, y)
            average = (r + g + b) / 3
            if average < 128:
                image.setPixel(x, y, color)
            else:
                image.setPixel(x, y, whitePixel)

def main():
    filename = input( "Enter the image file name: ")
    red = int(input("Enter the integer from 0 to 255 for red: "))
    green = int(input("Enter the integer from 0 to 255 for green: "))
    blue = int(input("Enter the integer from 0 to 255 for blue: "))
    image=Image(filename)
    posterize(image,(red,green,blue))
    image.draw()

if __name__ == "__main__":
    main()
```

8.1

```
A GUI-based tax calculator.

Computes and prints the total tax, given the income and
number of dependents (inputs), and a standard deduction of
$10,000, an exemption amount of $3,000, and a flat tax rate
of 20%.
"""

from breezypythongui import EasyFrame

class TaxCalculator(EasyFrame):
    """Application window for the tax calculator."""

    def __init__(self):
        """Sets up the window and the widgets."""
        EasyFrame.__init__(self, title = "Tax Calculator")

        # Label and field for the income
        # (self.incomeField)
        self.addLabel(text = "Gross income",row = 0, column = 0)
        self.incomeField = self.addFloatField(value = "0.0",row = 0,column = 1, width=20)

        # Label and field for the number of dependents
        # (self.depField)
        self.addLabel(text = "Dependents",row = 1, column = 0)
        self.depField = self.addIntegerField(value = "0",row = 1,column = 1,width=10)

         # The command button
        self.computeButton = self.addButton(text = "Compute",row = 2, column =0,columnspan =2,command = self.computeTax)

        # Label and field for the tax
        # (self.taxField)
        self.addLabel(text = "Total tax",row = 3, column = 0)
        self.taxField = self.addFloatField(value = "0.0",row = 3,column = 1,width=20,precision=2)

    # The event handler method for the button
    def computeTax(self):
        """Obtains the data from the input field and uses
        them to compute the tax, which is sent to the
        output field."""
        income=self.incomeField.getNumber()
        numDependents=self.depField.getNumber()
        exemptionAmount=3000.0
        standardDeduction=10000.0
        tax=(income-standardDeduction-(numDependents*exemptionAmount))*0.2
        self.taxField.setNumber(tax)


def main():
    TaxCalculator().mainloop()

if __name__ == "__main__":
    main()
```

7.5

```
from PIL import Image

import numpy

def posterize():

picture = Image.open("picture.png")

gray_scale = picture.convert('L')

dat = numpy.asarray(gray).copy()

dat[dat < 128] = 0 # Black

dat[dat >= 128] = 255 # White

imfile = Image.fromarray(dat)

imfile.save("result_dat.png")

posterize()
```

7.2

Unverified (ch 7 closed) found on the web

```
from turtlegraphics import Turtle
import random

def cCurve(turtle, x1, y1, x2, y2, level):

    """Draws a c-curve of the given level."""
    def drawLine(x1, y1, x2, y2):
        """Draws a line segment between the endpoints."""
        turtle.setColor(random.randint(0, 255), random.randint(0, 255), random.randint(0, 255))
        turtle.up()
        turtle.move(x1, y1)
        turtle.down()
        turtle.move(x2, y2)
    if level == 0:
        drawLine(x1, y1, x2, y2)
    else:
        xm = (x1 + x2 + y1 - y2) / 2
        ym = (x2 + y1 + y2 - x1) / 2

        cCurve(turtle, x1, y1, xm, ym, level - 1)

        cCurve(turtle, xm, ym, x2, y2, level - 1)
def main():
    level = input("Enter the level (0 or greater): ")
    turtle = Turtle(400, 500)
    turtle.setWidth(1)
    cCurve(turtle, 50, -100, 50, 100, level)

main()
```

6.6

```
"""
File: filesys.py
Project 6.6

Provides a menu-driven tool for navigating a file system
and gathering information on files.

Adds a command to view a file's contents.
"""

import os, os.path

QUIT = '8'

COMMANDS = ('1', '2', '3', '4', '5', '6', '7', '8')

MENU = """1   List the current directory
2   Move up
3   Move down
4   Number of files in the directory
5   Size of the directory in bytes
6   Search for a file name
7   View the contents of a file
8   Quit the program"""

def main():
    while True:
        print(os.getcwd())
        print(MENU)
        command = acceptCommand()
        runCommand(command)
        if command == QUIT:
            print("Have a nice day!")
            break

def acceptCommand():
    """Inputs and returns a legitimate command number."""
    while True:
        command = input(str("Enter a number: "))
        if not command in COMMANDS:
            print("Error: command not recognized")
        else:
            return command

def runCommand(command):
    """Selects and runs a command."""
    if command == '1':
        listCurrentDir(os.getcwd())
    elif command == '2':
        moveUp()
    elif command == '3':
        moveDown(os.getcwd())
    elif command == '4':
        print("The total number of files is", \
              countFiles(os.getcwd()))
    elif command == '5':
        print("The total number of bytes is", \
              countBytes(os.getcwd()))
    elif command == '6':
        target = raw_input("Enter the search string: ")
        fileList = findFiles(target, os.getcwd())
        if not fileList:
            print("String not found")
        else:
            for f in fileList:
                print(f)
    elif command == '7':
        viewFile(os.getcwd())

def viewFile(dirName):
    lyst = list(filter(os.path.isfile, os.listdir(dirName)))
    if len(lyst) == 0:
        print("There are no files in this directory")
    else:
        while True:
            print("Files in " + dirName + ":")
            for element in lyst: print(element)
            fileName = input("Enter a file name from these names: ")
            if not fileName in lyst:
                print("Sorry, there is an error in your file name.")
            else:
                f = open(fileName, 'r')
                print(f.read())
                break

def listCurrentDir(dirName):
    """Prints a list of the cwd's contents."""
    lyst = os.listdir(dirName)
    for element in lyst: print(element)

def moveUp():
    """Moves up to the parent directory."""
    os.chdir("..")

def moveDown(currentDir):
    """Moves down to the named subdirectory if it exists."""
    newDir = input("Enter the directory name: ")
    if os.path.exists(currentDir + os.sep + newDir) and \
       os.path.isdir(newDir):
        os.chdir(newDir)
    else:
        print("ERROR: no such name")

def countFiles(path):
    """Returns the number of files in the cwd and
    all its subdirectories."""
    count = 0
    lyst = os.listdir(path)
    for element in lyst:
        if os.path.isfile(element):
            count += 1
        else:
            os.chdir(element)
            count += countFiles(os.getcwd())
            os.chdir("..")
    return count

def countBytes(path):
    """Returns the number of bytes in the cwd and
    all its subdirectories."""
    count = 0
    lyst = os.listdir(path)
    for element in lyst:
        if os.path.isfile(element):
            count += os.path.getsize(element)
        else:
            os.chdir(element)
            count += countBytes(os.getcwd())
            os.chdir("..")
    return count

def findFiles(target, path):
    """Returns a list of the file names that contain
    the target string in the cwd and all its subdirectories."""
    files = []
    lyst = os.listdir(path)
    for element in lyst:
        if os.path.isfile(element):
            if target in element:
                files.append(path + os.sep + element)
        else:
            os.chdir(element)
            files.extend(findFiles(target, os.getcwd()))
            os.chdir("..")
    return files

if __name__ == "__main__":
    main()
```

6.1

```
# Modify the code below
"""
Program: newton.py
Author: Ken
Compute the square root of a number.
1. The input is a number.
2. The outputs are the program's estimate of the square root
   using Newton's method of successive approximations, and
   Python's own estimate using math.sqrt.
"""

import math

def newton(x):
    tolerance = 0.000001
    estimate =1
    while True:
        estimate = (estimate + x / estimate)/2
        difference = abs(x-estimate **2)
        if difference <= tolerance:
            break
    return float(estimate)

def main():
    x = 1
    while True:
        x = input("enter a positive number or enter/return to quit: ")
        if x == "":
            break
        else:
            x=float(x)
        print("The program's estimate is", newton(x))
        print("Python's estimate is      ", math.sqrt(x))
if __name__ =="__main__":
    main()
```

Posterize

```
"""
File: testbandw.py
Tests a function for converting a color image to
black and white.
"""

from images import Image

def posterize(image,color=(0,0,0)):
    """Convert a color image to two color, one of which is
white and the other which wither a default of black or user specified RGB value"""
    blackPixel = (0, 0, 0)
    whitePixel = (255, 255, 255)
    for y in range(image.getHeight()):
        for x in range(image.getWidth()):
            (r, g, b) = image.getPixel(x, y)
            average = (r + g + b) / 3
            if average < 128:
                image.setPixel(x, y, color)
            else:
                image.setPixel(x, y, whitePixel)

def main():
    filename =input("Enter the image file name: ")
    red=int(input("Enter an integer value from 0 to 255 for red: "))
    green=int(input("Enter an integer value from 0 to 255 for green: "))
    blue=int(input("Enter an integer value from 0 to 255 for blue: "))
    image = Image(filename)
    posterize(image,(red,green,blue))
    image.draw()
    image.draw()

if __name__ == "__main__":
   main()
```

Programming Exercise 7.3

```
from turtle import Turtle, tracer, update
import math
import sys

def drawKochFractal(width, height, size, level):
    """draw a Koch fractal of the given level and size"""
    t=Turtle()
    t.hideturtle()
    t.up()
    t.goto(-width//3, height//4)
    t.down()
    t.speed(0)
    drawFractalLine(t, size, 0, level)
    drawFractalLine(t, size, -120, level)
    drawFractalLine(t, size, 120, level)

def drawFractalLine(t, distance, theta, level):
    """Draw a single line in a direction or 4 fractal lines in new direction"""
    if(level==0):
        drawPolarLine(t, distance, theta)
    else:
        drawFractalLine(t, distance//3, theta, level - 1)
        drawFractalLine(t, distance//3, theta + 60, level - 1)
        drawFractalLine(t, distance//3, theta - 60, level - 1)
        drawFractalLine(t, distance//3, theta, level - 1)

def drawPolarLine(t, distance, theta):
    """Move a given distance in the direction"""
    t.setheading(theta)
    t.forward(distance)
```

```
def main():
    drawKochFractal(200,200, 150, 4)

if __name__=="__main__":
    main()
```

7.1

```
import math
from turtle import Turtle

def drawCircle(t,x,y,radius):
    t.up()
    t.goto(x + radius, y)
    t.setheading(90)
    t.down()
    for count in range(120):
        t.left(3)
        t.forward(2.0 * math.pi * radius/120.0)

def main():
        x=25
        y=75
        radius = 100
        drawCircle(Turtle(),x,y,radius)

if __name__ =="__main__":
    main()
```

CH 6 QUIZ

- Top-down design is a strategy that: starts with the main function and develops the functions on each successive level beneath the main function
- When a recursive function is called, the values of its arguments and its return address are placed in a: stack frame
- The scope of a temporary variable is: the statements in the body of the function after the statement where the variable is introduced
- The relationships among functions in a top-down design are shown in a(n) Structure chart
- A recursive function: usually runs more slowly than the equivalent loop
- The expression list(filter(lambda x: x > 50, [34, 65, 10, 100])) evaluates to: [65,100]
- The lifetime of a parameter is: the duration of its function's execution
- The expression list(map(math.sqrt, [9, 25, 36])) evaluates to :[3.0, 5.0, 6.0]
- The expression reduce(max, [34, 21, 99, 67, 10]) evaluates to: 99
- A data structure used to implement a jump table is a: Dictionary
- The values of an object's instance variables make up its state. T
-

Unit 5 Quiz

- The variable data refers to the list [10, 20, 30]. The expression data[1:3] evaluates to: [20,30]
- The variable data refers to the list [10, 20, 30]. The expression data.index(20) evaluates to: 1
- The variable data refers to the list [10, 20, 30]. After the statement data[1] = 5, data evaluates to: [10, 5, 30]
- The variable info refers to the dictionary {"name":"Sandy", "age":17}.
- The method to remove an entry from a dictionary is named: pop
- Which of the following are immutable data structures? Strings and tuples
- The variable data refers to the list [10, 20, 30]. The expression data[1] evaluates to: 20
- The variable info refers to the dictionary {"name":"Sandy", "age":17}.
- The expression info.get("hobbies", None) evaluates to None
- The variable info refers to the dictionary {"name":"Sandy", "age":17}.
- The expression list(info.keys()) evaluates to: ["name", "age"]
- The variable data refers to the list [10, 20, 30]. The expression data + [40, 50] evaluates to: [10, 20, 30, 40, 50]
- The variable data refers to the list [10, 20, 30]. After the statement data.insert(1, 15), the original data evaluates to:
- [10, 15, 20, 30]

Midterm Practice Exam

- Which of the following literals would be considered a float type in Python? 3.14
- A flash memory stick is an example of what type of storage media?semiconductor storage media
- The string is a mutable data structure. False
- You are running a Python script that is supposed to write data to an open file, but discover that the file contents are being erased each time the script runs. What is most likely the issue?The file was opened using a mode string other than 'a'.
- Information is stored as patterns of bytes (1s and 0s) False
- What os module function should you use to get the path of the current working directory?getcwd()
- Given the following statement: "%4d" % var1, which of the following is accurate?The variable var1 must be an integer value.
- Simple Boolean expressions consist of the Boolean values True or False, variables bound to those values, function calls that return Boolean values, or comparisons. True
- After input is finished, another call to read returns an empty string to indicate that the end of the file has been reached. True
- An algorithm describes a process that ends with a solution to a problem. True
- The statements within a while loop can execute one or more times. False
- The statements in the loop body need not be indented and aligned in the same column. False
- The decimal number system, which utilizes the numbers 0-9, is also known as what number system? The base-10 numbering system
- When using the range function, what effect does the specification of a third argument have?The third argument specifies a step value to be used in the range.
- There are two types of loops—those that repeat an action a predefined number of times (definite iteration) and those that perform the action until the program determines that it needs to stop (indefinite iteration).True
- The use of +=, -=, and *= are all examples of what type of operators? augmented assignment operators
- What does a mode string of 'r' indicate when opening a file in Python?The file will be opened for read access.
- A while loop can be used for a count-controlled loop. True
- When the Python interpreter evaluates a literal, the value it returns is simply that literal, True
- What is the decimal number 98 in binary? 1100010
- What is a count controlled loop? It is a loop that counts through a range of numbers to control when the loop ends.
- You are parsing a comma separated value string with Python. What method can you use to separate the values in each line and place them in a list?string.split(',')
- What function call will generate a random number in the range of 1 through 6 using the random module? random.randint(1, 6)
- The comparison operators are applied after addition but before assignment. True
- What function can you call inside of IDLE to show the resources of the math module, once it has been imported?dir(math)
- string is an example of a data type in Python. False
- Conditional iteration requires that a condition be tested within the loop to determine whether the loop should continue. True
- Modern software development is usually incremental and iterative. True
- What scenario would it make logical sense to use a multi-way selection statement for?You need to convert numeric grades to letter grades.
- When using the open function to open a file for output (i.e., using 'w' as the mode string), if the file does not exist, Python raises an error. False
- Functions that are always available in Python come from what module? __builtin__
- A variable associates a name with a value, making it easy to remember and use the value later in a program.True
- You can use parentheses to change the order of evaluation in an arithmetic expression. True
- What statement should you use to print the value of total with a precision of 5 digits?print("The total is %0.5f" % total)
- Assuming that the value of x is 4, and the value of y is 6, what is the value of the following expression? (x - y)**2 > y : False
- 4 != 4 evaluates to True. False
- What character is used as the format operator when formatting output? %
- Some applications extract portions of strings called characters. False
- What print statement will output a single '\' character?print('\\')
- What happens if you attempt to access a string using an index equal to the string's length?An error will occur, indicating the index is out of range
- In Python, what does the "%" operator do in the expression 6 % 4? It returns a remainder or modulus
- What specific Python module can you use to determine the file size in bytes of a given file?os.path
- 1,500 is a valid integer literal in Python. False

- Which of the following is NOT a valid name that can be used for a variable?1ending
- In Python, = means equals, whereas == means assignment. False
- What is the total number of distinct values in the ASCII set? 256
- When using the open function to open a file for input (i.e., using 'r' as the mode string), if the file does not exist, Python raises an error. T
- Which of the following are examples of protocols that utilize encryption for data transfer? (Choose two.) FTPS, HTTPS
- The decimal number system is also called the base ten number system. True
- The two digits in the base two number system are 0 and 1. True
- What statement should you use to print the value of total with a precision of 5 digits? print("The total is %0.5f" % total)
-

5.2

```
fileName=input("Enter the input file name: ")
file=open(fileName).readlines()
count = len(file)
print()

while True:
        print("The file has", count, "lines.")
        lineNumber = input("Enter a line number [0 to quit]: ")
        if float(lineNumber)>0:
        print(lineNumber,": ",file[eval(lineNumber)-1])
        elif float(lineNumber)==0:
        break
```

5.1

- page 147

```
from statistics import mode
numbers = [3, 1, 7, 1, 4, 10]
numbers.sort()
print("List:",str(numbers))
print("Mode:",mode(numbers))
median = len(numbers)//2
if len(numbers) % 2 == 1:
   print(numbers[median])
else:
   print("Median:",(numbers[median] + numbers[median -1])/2)
counter = 0
for index in range(len(numbers)):
   counter+=numbers[index]
print("Mean:",counter/len(numbers))
```

4.12

```
enteredName =open(input("Enter the file name: "), "r")
count = 0

print( "%-12s%10s%12s"%("Name","Hours", "TotalPay"))
while True:
   line = enteredName.readline()

   if line == "":
      break
   words = line.split()
   hours = words[1]
   wage = words[2]
   hourNum = float(hours)
   wageNum = float(wage)
   print("%-12s%10s%12.2f"%((words[0]), hours, (hourNum * wageNum)))
```

4.8

```
copyFrom = input("Enter the input file name: ")
copyTo = input("Enter the output file name: ")

f = open(copyFrom,"r")
text= f.read()
f = open(copyTo,"w")
f.write(text)
```

4.1

```
plainText = str(input("Enter a message:"))
distance = int(input("Enter the distance value:"))
code = ""
for ch in plainText:
    ordvalue = ord(ch)
    cipherValue = ordvalue + distance
    if cipherValue > 127:
        cipherValue = distance - (127 - ordValue +1)
    code += chr(cipherValue)
print(code)
```

4.2

```
plainText=input("Enter the coded text: ")
distance=int(input("Enter the distance value: "))
code=""
for ch in plainText:
    ordvalue=ord(ch)
    cipherValue=ordvalue - distance
    if cipherValue > 127:
        cipherValue = distance - (127 - ordValue +1)
    code += chr(cipherValue)
print(code)
```

OR

```
code=input("Enter the coded text: ")
distance=int(input("Enter the distance value: "))
plainText=""
for ch in code:
    ordvalue=ord(ch)
    cipherValue=ordvalue - distance
    if cipherValue < 0:
        cipherValue = 127 - (distance - (1- ordValue)
    plainText += chr(cipherValue)
print(plainText)
```

4.3

```
#######################decrypt.py########################
filename=(input("Enter the input file name: "))
outputname=(input("Enter the output file name: "))
f=open(filename,"r")
code = f.read()
distance = int(input("Enter the distance value: "))
plainText = ''
for ch in code:
	ordValue = ord(ch)
	cipherValue = ordValue - distance
	if cipherValue > 127:
	cipherValue = distance - (127 - ordValue+1)
	plainText +=  chr(cipherValue)
f.close()
f=open(outputname,"w")
f.write(plainText)
f.close()
#######################decrypt.py########################

#######################encrypt.py########################
filename=(input("Enter the input file name: "))
outputname=(input("Enter the output file name: "))
f=open(filename,"r")
code = f.read()
distance = int(input("Enter the distance value: "))
plainText = ''
for ch in code:
	ordValue = ord(ch)
	cipherValue = ordValue + distance
	if cipherValue > 127:
	cipherValue = distance - (127 - ordValue+1)
	plainText +=  chr(cipherValue)
f.close()
f=open(outputname,"w")
f.write(plainText)
f.close()
#######################encrypt.py########################
```

4.10

```
firstFile = open(input("Enter the first file name: "), "r")
secondFile = open(input("Enter the second file name: "), "r")

line1 = firstFile.readlines()
line2 = secondFile.readlines()

if line1 == line2:
    print("\nYes.")

for line in range(0, min(len(line1), len(line2))):
    if line1[line] != line2[line]:
        print("\n No\n", line1[line], line2[line])
```

3.3

```
# Modify the code below:

import math

smaller = int(input("Enter the smaller number: "))
larger = int(input("Enter the larger number: "))
count = 0
maxGuesses=round(math.log(larger-smaller+1,2))
while True:
    count += 1
    print(smaller, larger)
    yourNumber=(smaller+larger)//2
    print("Your number is",yourNumber)
    answer=input("enter =, <, or >")
    if answer == "=":
        print("Hooray, I've got it in",count,"tries!")
        break
    elif count==maxGuesses:
        print("I'm out of guesses, and you cheated!")
        break
    elif answer=="<":
        larger=yourNumber-1
    else:
        smaller=yourNumber+1
```