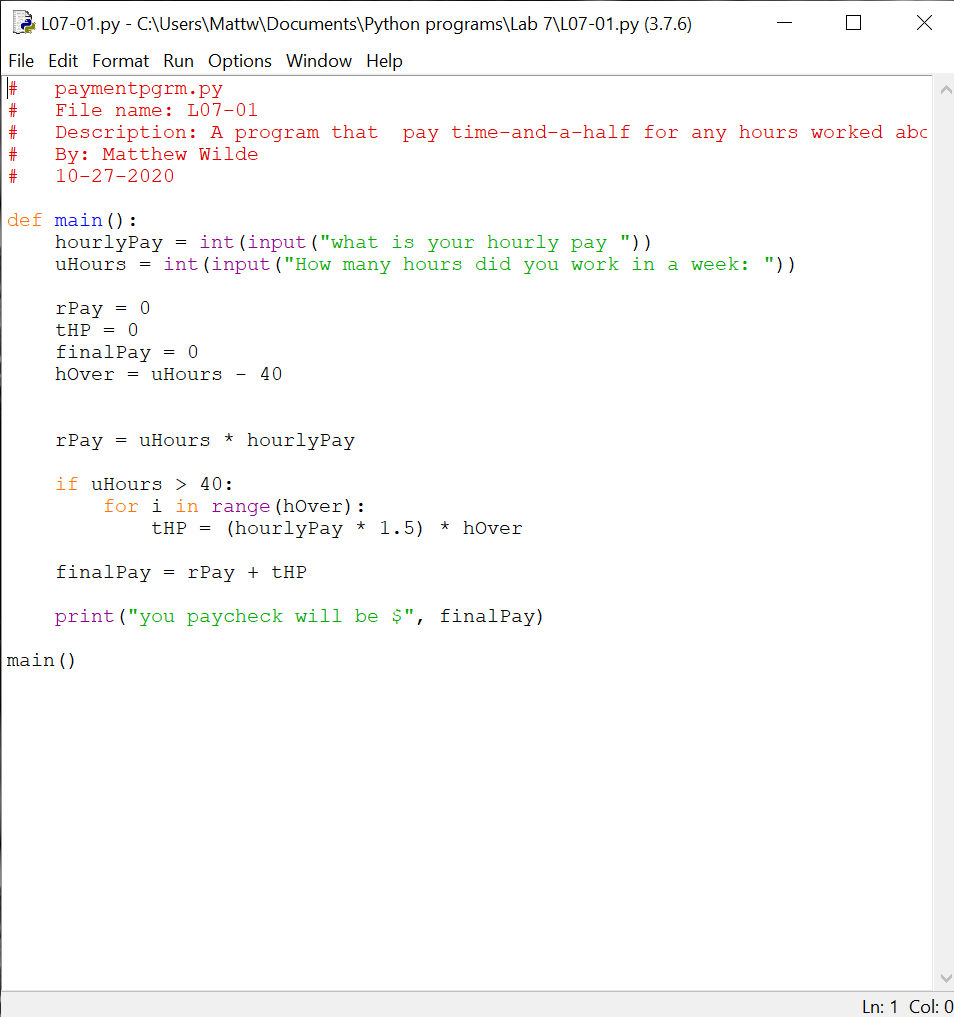
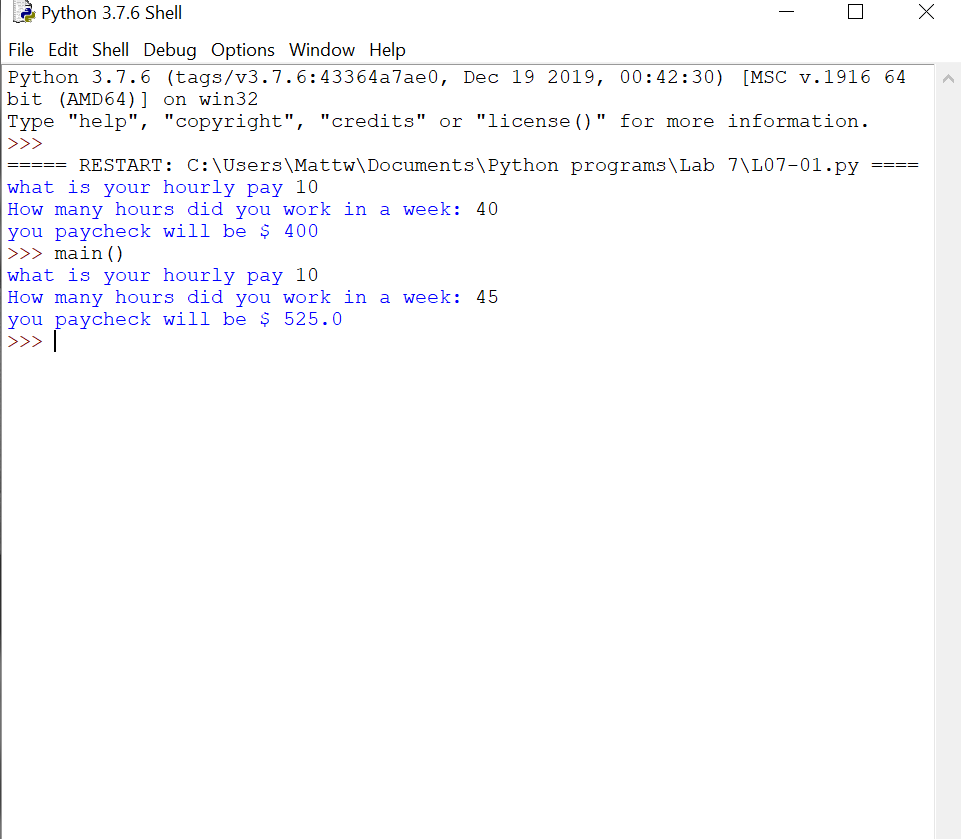
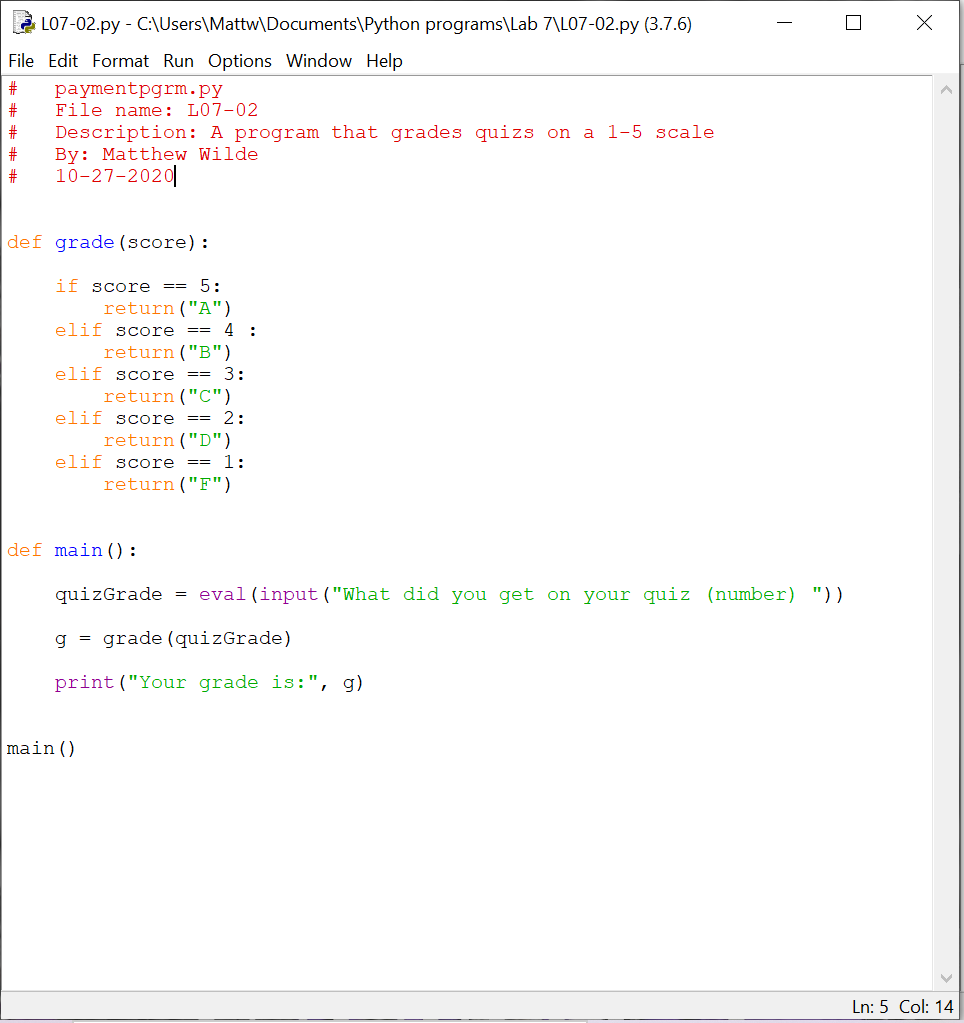
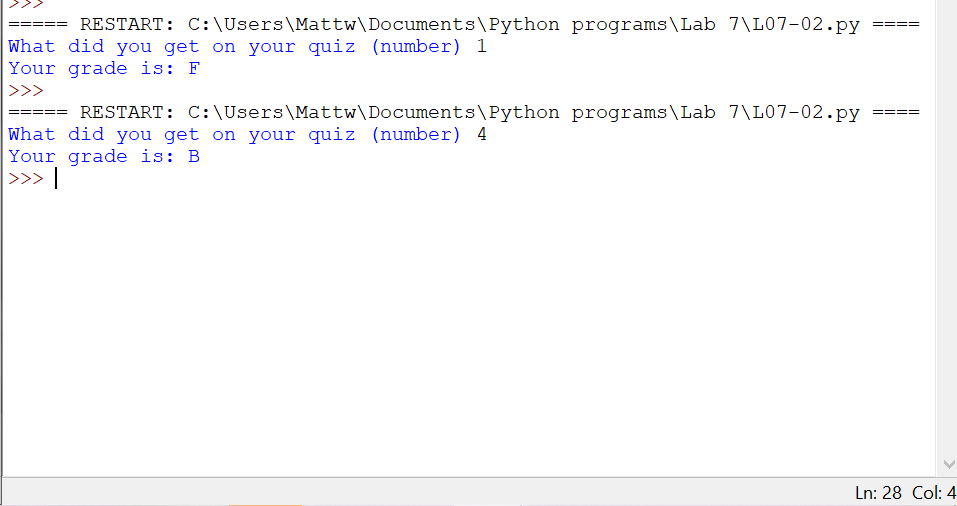
Matthew Wilde  
101-1 Lab 07, pages 239 - 241  
   
**Question CH.07PE:1**  
Many companies pay time-and-a-half for any hours worked above 40 in a given week. Write a program to input the number of hours worked and the hourly rate and calculate the total wages for the week.

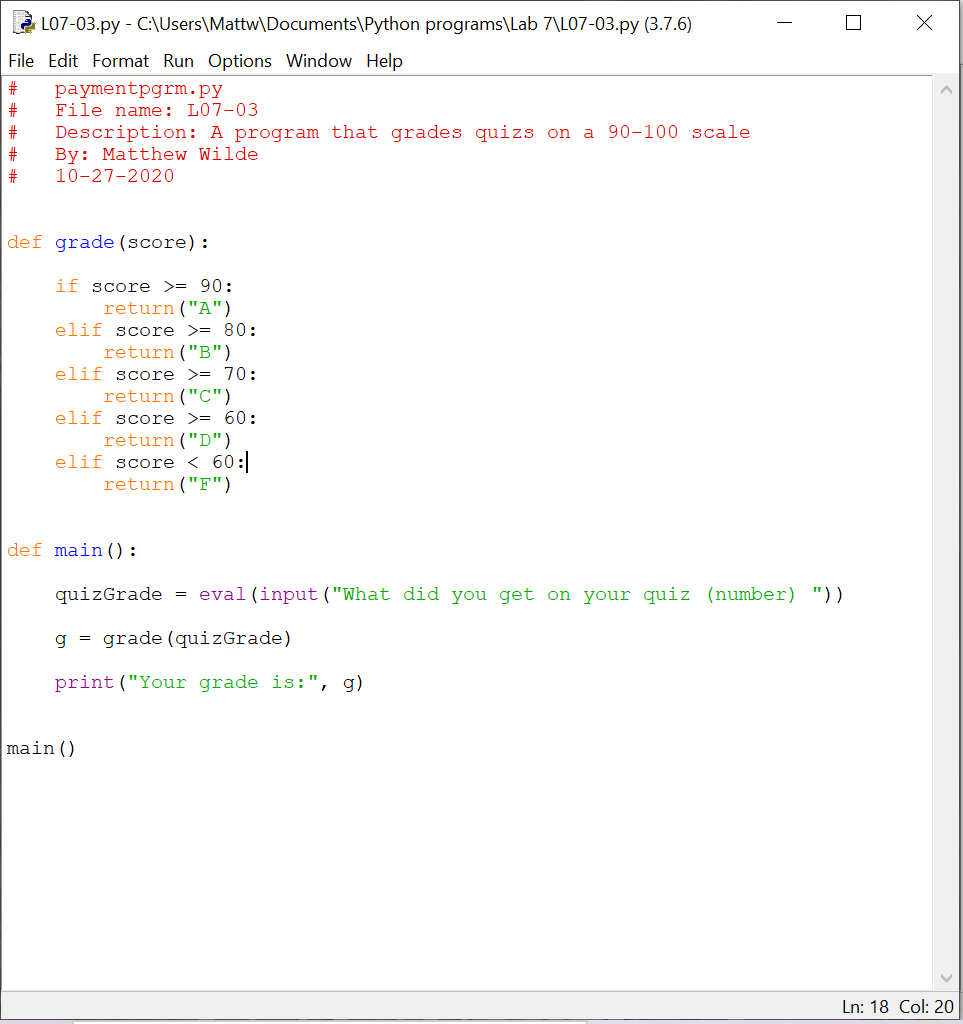
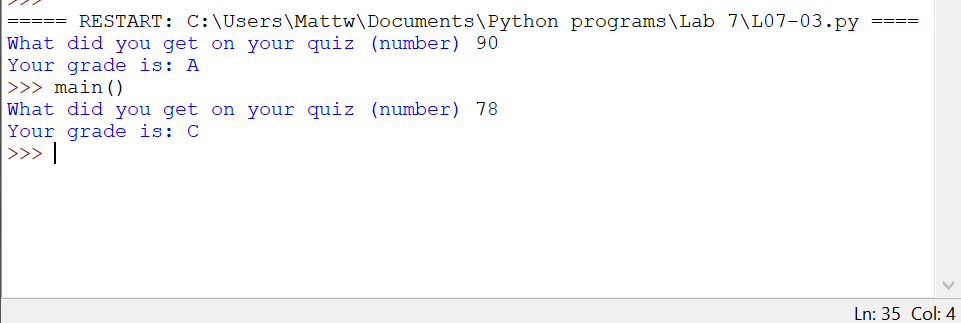
**Written Code:**# paymentpgrm.py  
# File name: L07-01  
# Description: A program that pay time-and-a-half for any hours worked above 40 in a given week.  
# By: Matthew Wilde  
# 10-27-2020  
  
def main():  
 hourlyPay = int(input("what is your hourly pay "))  
 uHours = int(input("How many hours did you work in a week: "))   
   
 rPay = 0  
 tHP = 0  
 finalPay = 0  
 hOver = uHours - 40  
  
  
 rPay = uHours \* hourlyPay  
  
 if uHours > 40:  
 for i in range(hOver):  
 tHP = (hourlyPay \* 1.5) \* hOver  
  
 finalPay = rPay + tHP  
  
 print("you paycheck will be $", finalPay)  
   
main()

**Screenshots of Code:** **Screenshots of Running Results:**  
**Question CH.07PE:2**  
A certain CS professor gives five-point quizzes that are graded on the scale 5-A, 4-B, 3-C, 2-D, 1-F, 0-F. Write a program that accepts a quiz score as an input and uses a decision structure to calculate the corresponding grade.

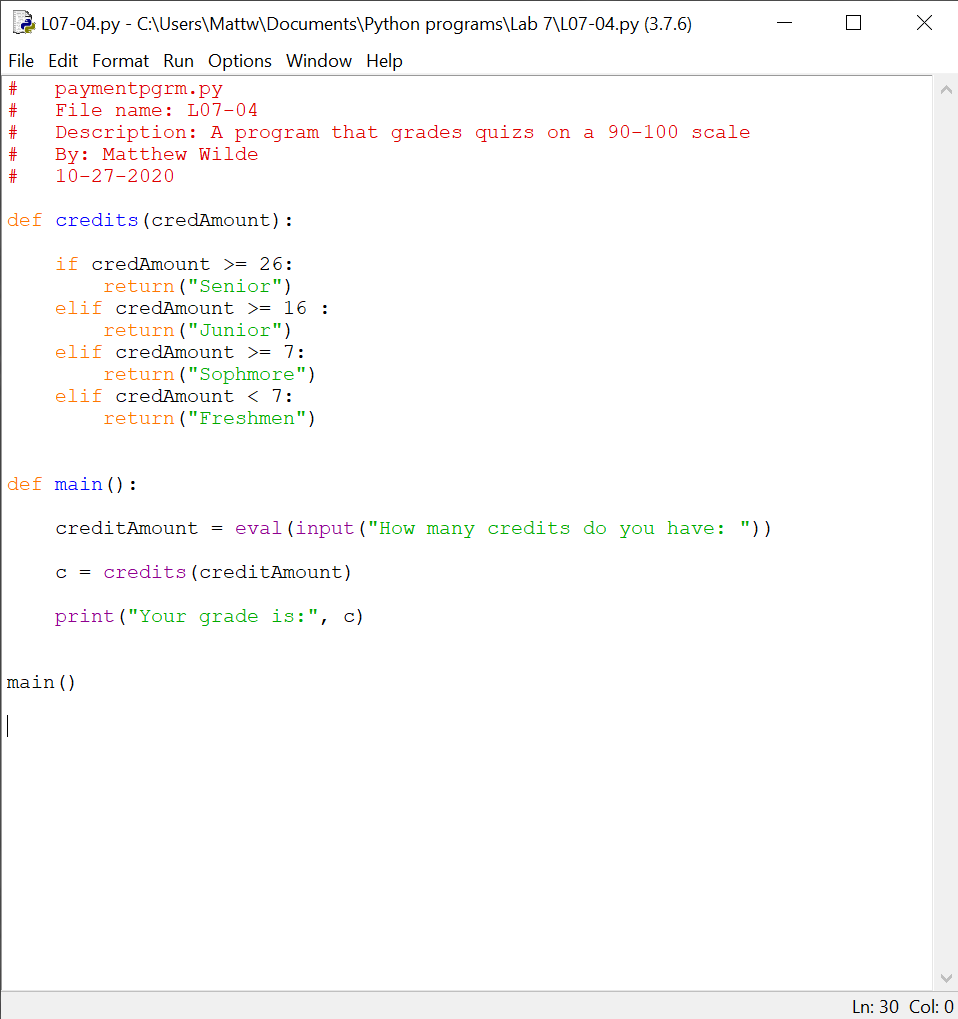
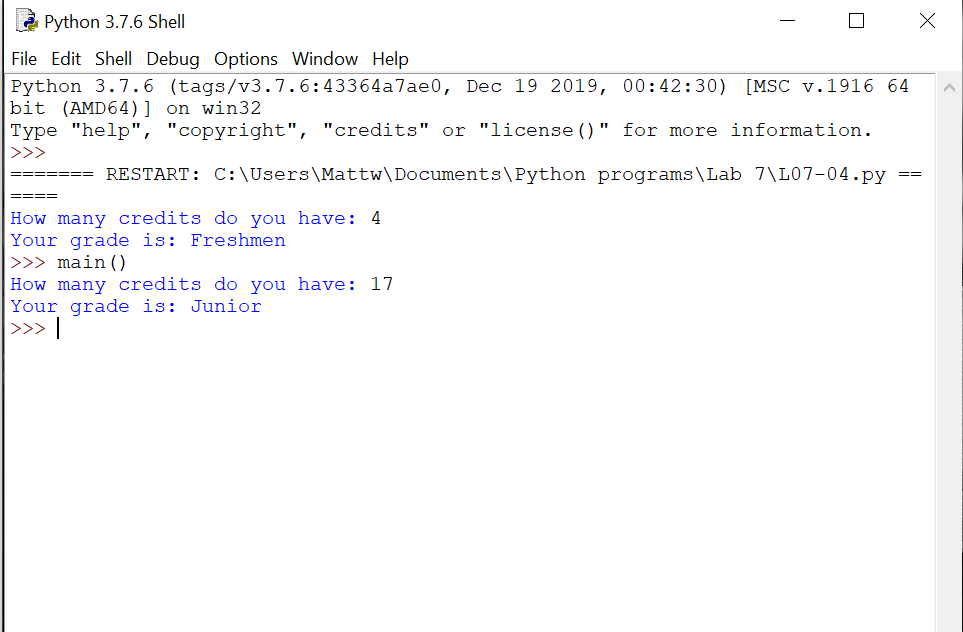
**Written Code:**# paymentpgrm.py  
# File name: L07-02  
# Description: A program that grades quizs on a 1-5 scale  
# By: Matthew Wilde  
# 10-27-2020  
  
  
def grade(score):  
   
 if score == 5:  
 return("A")  
 elif score == 4 :  
 return("B")  
 elif score == 3:  
 return("C")  
 elif score == 2:  
 return("D")  
 elif score == 1:  
 return("F")  
   
  
def main():  
   
 quizGrade = eval(input("What did you get on your quiz (number) "))  
  
 g = grade(quizGrade)  
  
 print("Your grade is:", g)  
  
  
main()

**Screenshots of Code:** **Screenshots of Running Results:**  
**Question CH.07PE:3**  
A certain CS professor gives 100-point exams that are graded on the scale 90-100:A, 80-89:B, 70-79:C, 60-69:0, <60:F. Write a program that accepts an exam score as input and uses a decision structure to calculate the corresponding grade.

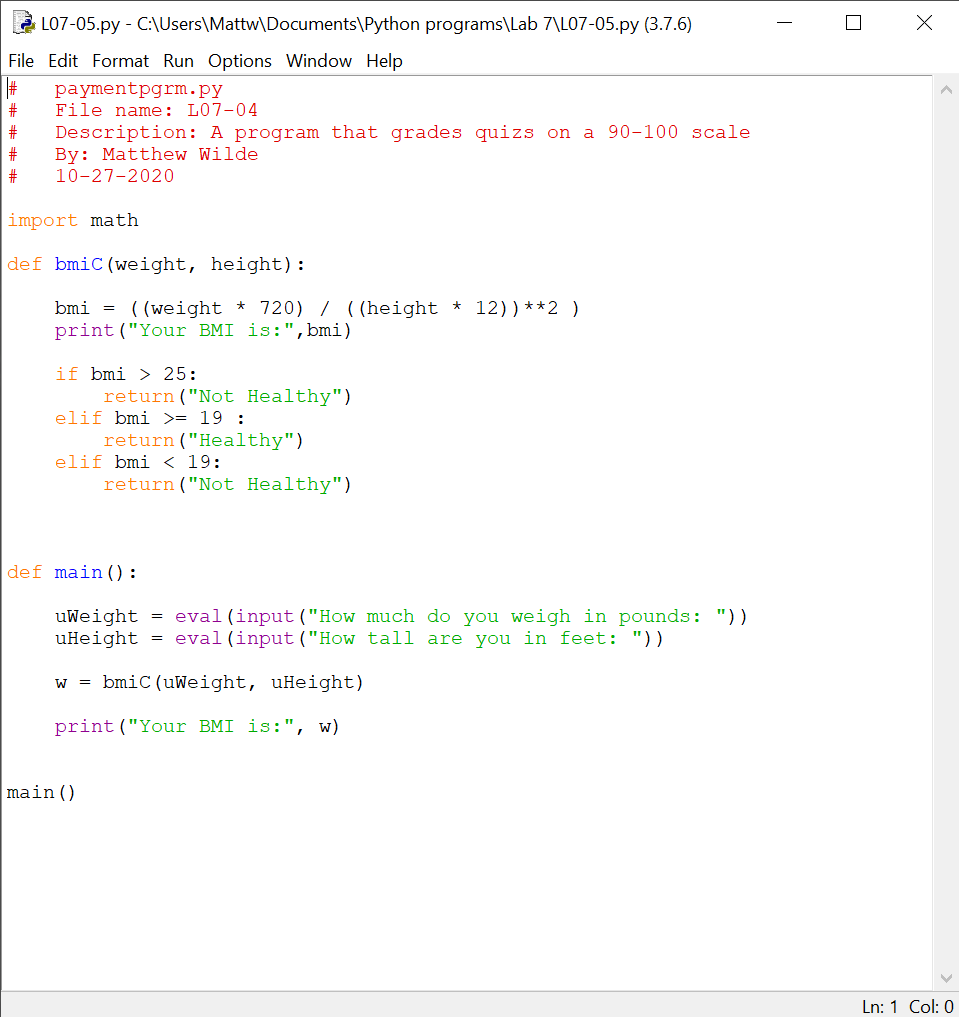
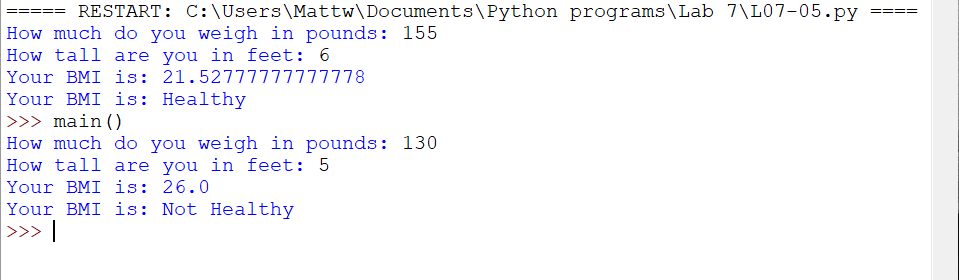
**Written Code:**# paymentpgrm.py  
# File name: L07-03  
# Description: A program that grades quizs on a 90-100 scale  
# By: Matthew Wilde  
# 10-27-2020  
  
  
def grade(score):  
   
 if score >= 90:  
 return("A")  
 elif score >= 80:  
 return("B")  
 elif score >= 70:  
 return("C")  
 elif score >= 60:  
 return("D")  
 elif score < 60:  
 return("F")  
   
  
def main():  
   
 quizGrade = eval(input("What did you get on your quiz (number) "))  
  
 g = grade(quizGrade)  
  
 print("Your grade is:", g)  
  
  
main()

**Screenshots of Code:** **Screenshots of Running Results:**  
**Question CH.07PE:4**  
A certain college classifies students according to credits earned. A student with less than 7 credits is a Freshman. At least 7 credits are required to be a Sophomore, 16 to be a Junior and 26 to be classified as a Senior. Write a program that calculates class standing from the number of credits earned.

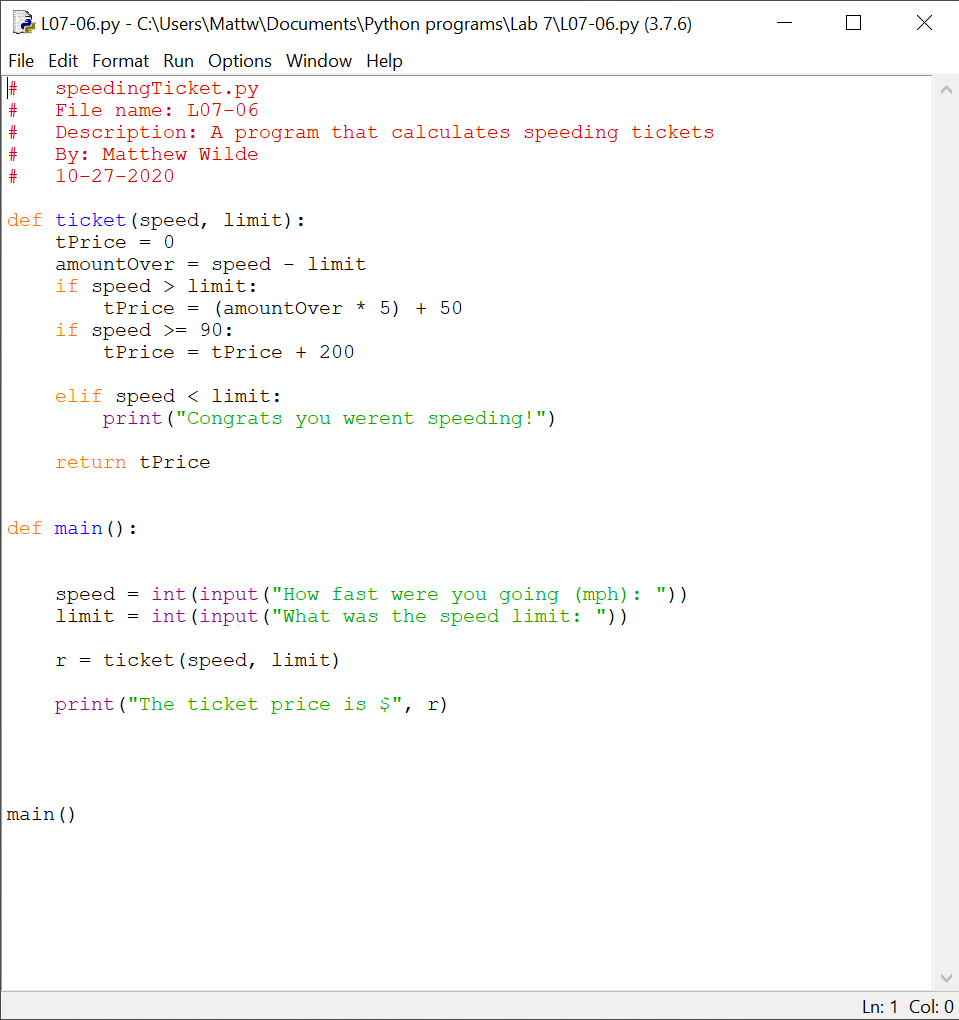
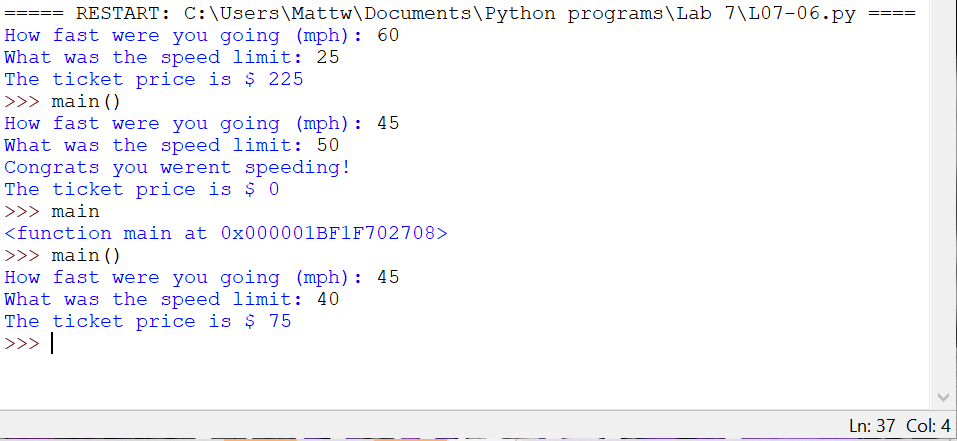
**Written Code:**# paymentpgrm.py  
# File name: L07-04  
# Description: A program that grades quizs on a 90-100 scale  
# By: Matthew Wilde  
# 10-27-2020  
  
def credits(credAmount):  
   
 if credAmount >= 26:  
 return("Senior")  
 elif credAmount >= 16 :  
 return("Junior")  
 elif credAmount >= 7:  
 return("Sophmore")  
 elif credAmount < 7:  
 return("Freshmen")  
   
  
def main():  
   
 creditAmount = eval(input("How many credits do you have: "))  
  
 c = credits(creditAmount)  
  
 print("Your grade is:", c)  
  
  
main()

**Screenshots of Code:** **Screenshots of Running Results:**  
**Question CH.07PE:5**  
The body mass index (BMI) is calculated as a person's weight (in pounds) times 720, divided by the square of the person's height (in inches). A BMI in the range 19-25, inclusive, is considered healthy. Write a program that calculates a person's BMI and prints a message telling whether they are above, within, or below the healthy range.

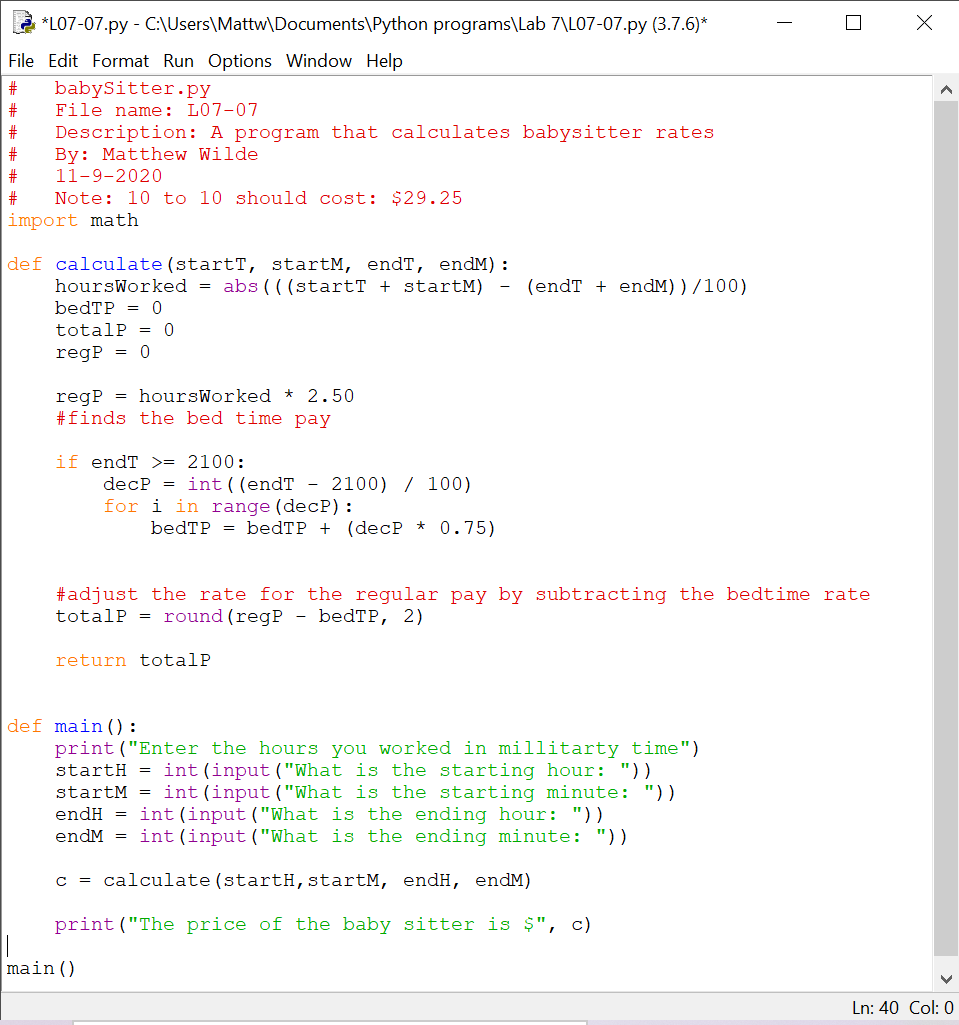
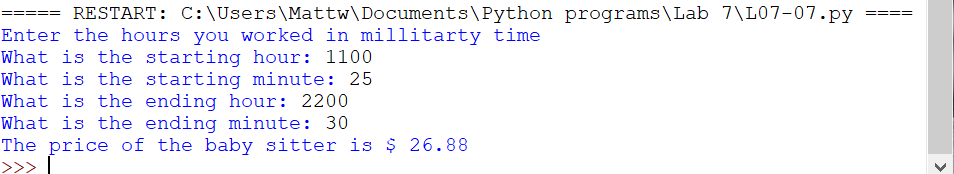
**Written Code:**# paymentpgrm.py  
# File name: L07-04  
# Description: A program that grades quizs on a 90-100 scale  
# By: Matthew Wilde  
# 10-27-2020  
  
import math  
  
def bmiC(weight, height):  
  
 bmi = ((weight \* 720) / ((height \* 12))\*\*2 )  
 print("Your BMI is:",bmi)  
   
 if bmi > 25:  
 return("Not Healthy")  
 elif bmi >= 19 :  
 return("Healthy")  
 elif bmi < 19:  
 return("Not Healthy")  
   
   
  
def main():  
   
 uWeight = eval(input("How much do you weigh in pounds: "))  
 uHeight = eval(input("How tall are you in feet: "))  
  
 w = bmiC(uWeight, uHeight)  
  
 print("Your BMI is:", w)  
  
  
main()

**Screenshots of Code:** **Screenshots of Running Results:**  
**Question CH.07PE:6**  
The speeding ticket fine policy in Podunksville is $50 plus $5 for each mph over the limit plus a penalty of $200 for any speed over 90 mph. Write a program that accepts a speed limit and a clocked speed and either prints a message indicating the speed was legal or prints the amount of the fine, if the speed is illegal.

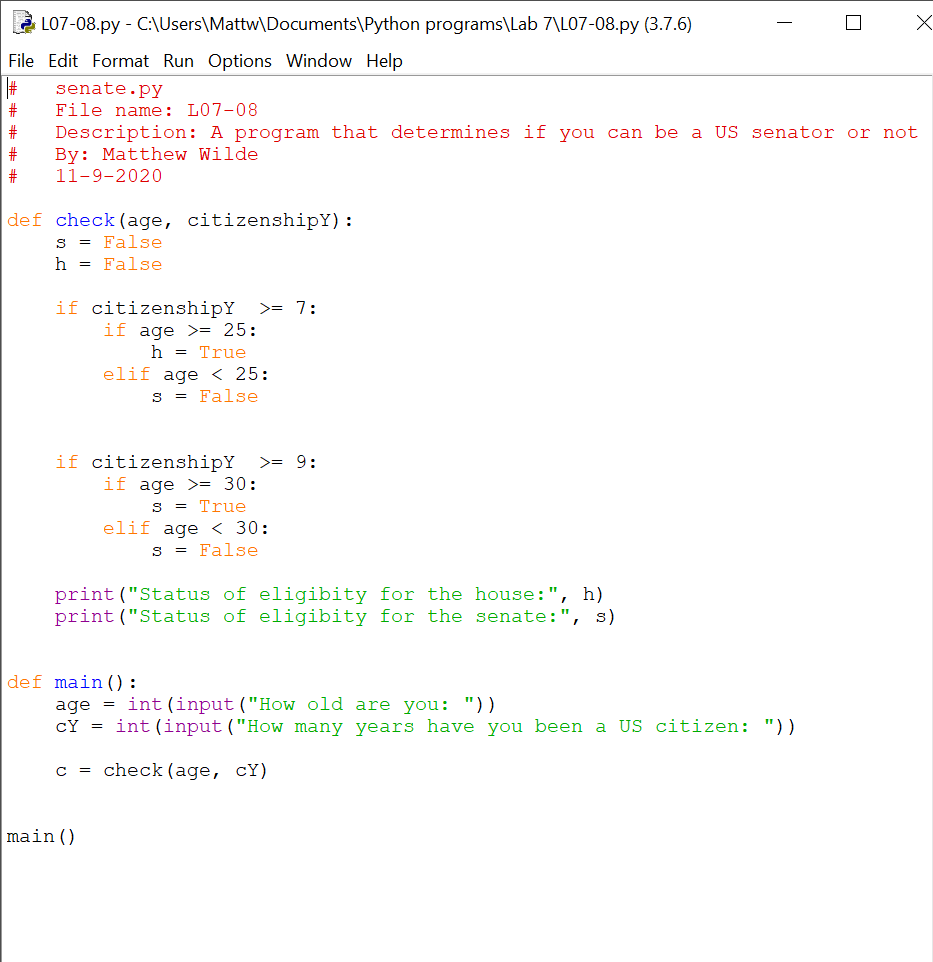
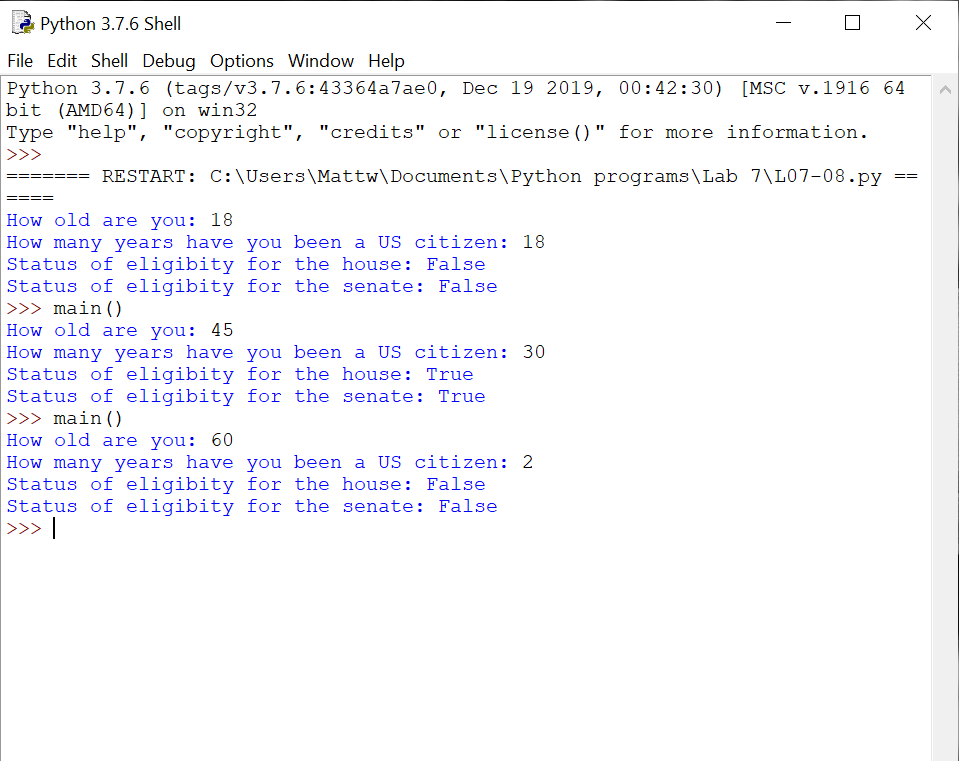
**Written Code:**# speedingTicket.py  
# File name: L07-06  
# Description: A program that calculates speeding tickets  
# By: Matthew Wilde  
# 10-27-2020  
  
def ticket(speed, limit):  
 tPrice = 0  
 amountOver = speed - limit  
 if speed > limit:  
 tPrice = (amountOver \* 5) + 50  
 if speed >= 90:  
 tPrice = tPrice + 200  
  
 elif speed < limit:  
 print("Congrats you werent speeding!")  
   
 return tPrice  
  
  
def main():  
   
  
 speed = int(input("How fast were you going (mph): "))  
 limit = int(input("What was the speed limit: "))  
  
 r = ticket(speed, limit)  
  
 print("The ticket price is $", r)  
   
  
  
  
main()

**Screenshots of Code:** **Screenshots of Running Results:**  
**Question CH.07PE:7**  
A babysitter charges $2.50 an hour until 9:00 PM when the rate drops to $1.75 an hour (the children are in bed). Write a program that accepts a starting time and ending time in hours and minutes and calculates the total babysitting bill. You may assume that the starting and ending times are in a single 24-hour period. Partial hours should be appropriately prorated.

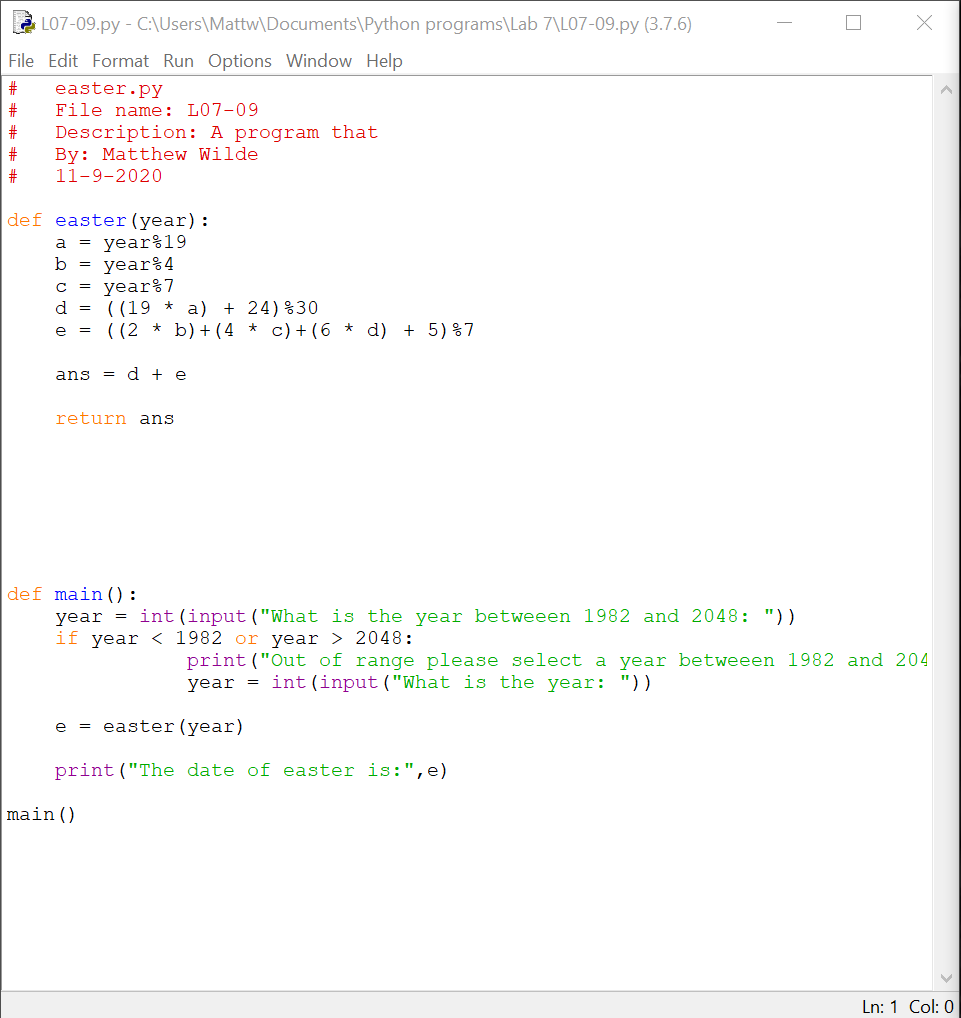
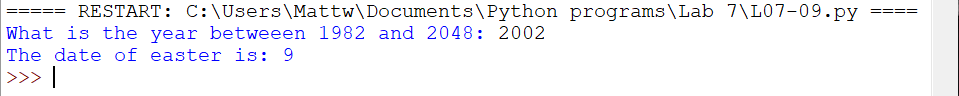
**Written Code:**# babySitter.py  
# File name: L07-07  
# Description: A program that calculates babysitter rates  
# By: Matthew Wilde  
# 11-9-2020  
# Note: 10 to 10 should cost: $29.25  
import math   
  
def calculate(startT, startM, endT, endM):  
 hoursWorked = abs(((startT + startM) - (endT + endM))/100)  
 bedTP = 0  
 totalP = 0  
 regP = 0   
   
 regP = hoursWorked \* 2.50  
 #finds the bed time pay  
   
 if endT >= 2100:  
 decP = int((endT - 2100) / 100)  
 for i in range(decP):  
 bedTP = bedTP + (decP \* 0.75)  
   
   
 #adjust the rate for the regular pay by subtracting the bedtime rate  
 totalP = round(regP - bedTP, 2)  
  
 return totalP  
   
  
def main():  
 print("Enter the hours you worked in millitarty time")  
 startH = int(input("What is the starting hour: "))  
 startM = int(input("What is the starting minute: "))  
 endH = int(input("What is the ending hour: "))  
 endM = int(input("What is the ending minute: "))  
  
 c = calculate(startH,startM, endH, endM)  
  
 print("The price of the baby sitter is $", c)  
  
main()

**Screenshots of Code:** **Screenshots of Running Results:**  
**Question CH.07PE:8**  
A person is eligible to be a US senator if they are at least 30 years old and have been a US citizen for at least 9 years. To be a US representative these numbers are 25 and 7, respectively. Write a program that accepts a person's age and years of citizenship as input and outputs their eligibility for the Senate and House.

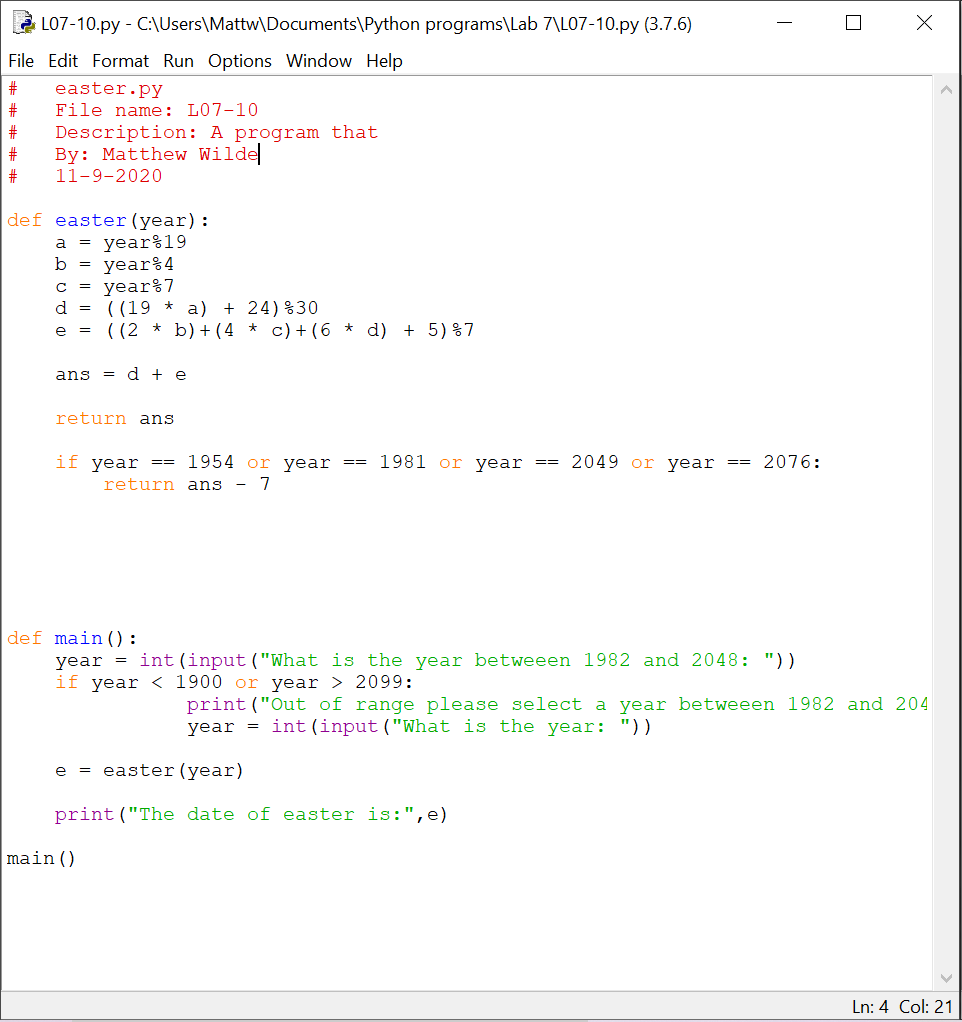
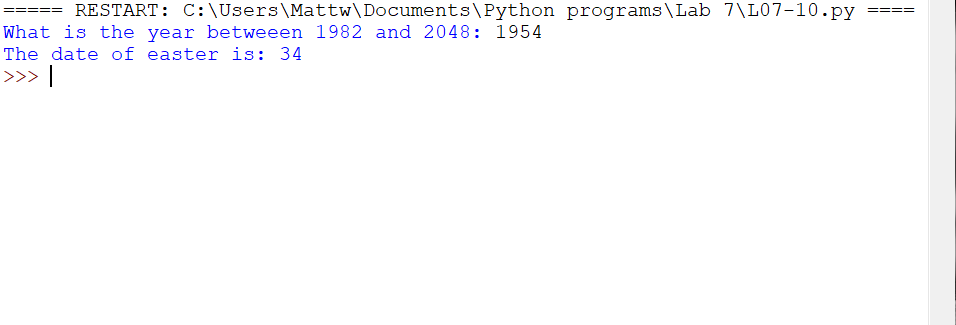
**Written Code:**# senate.py  
# File name: L07-08  
# Description: A program that determines if you can be a US senator or not  
# By: Matthew Wilde  
# 11-9-2020  
  
def check(age, citizenshipY):  
 s = False  
 h = False  
   
 if citizenshipY >= 7:  
 if age >= 25:  
 h = True  
 elif age < 25:  
 s = False  
   
   
 if citizenshipY >= 9:  
 if age >= 30:  
 s = True  
 elif age < 30:  
 s = False  
  
 print("Status of eligibity for the house:", h)  
 print("Status of eligibity for the senate:", s)  
  
  
def main():  
 age = int(input("How old are you: "))  
 cY = int(input("How many years have you been a US citizen: "))  
  
 c = check(age, cY)  
   
  
main()

**Screenshots of Code:** **Screenshots of Running Results:**  
**Question CH.07PE:9**  
A formula for computing Easter in the years 1982-2048, inclusive, is as follows: let a = year%19, b = year%4, c = year%7, d = (19a + 24)%30, e = (2b + 4c + 6d + 5)%7. The date of Easter is March 22 + d + e (which could be in April). Write a program that inputs a year, verifies that it is in the proper range, and then prints out the date of Easter that year

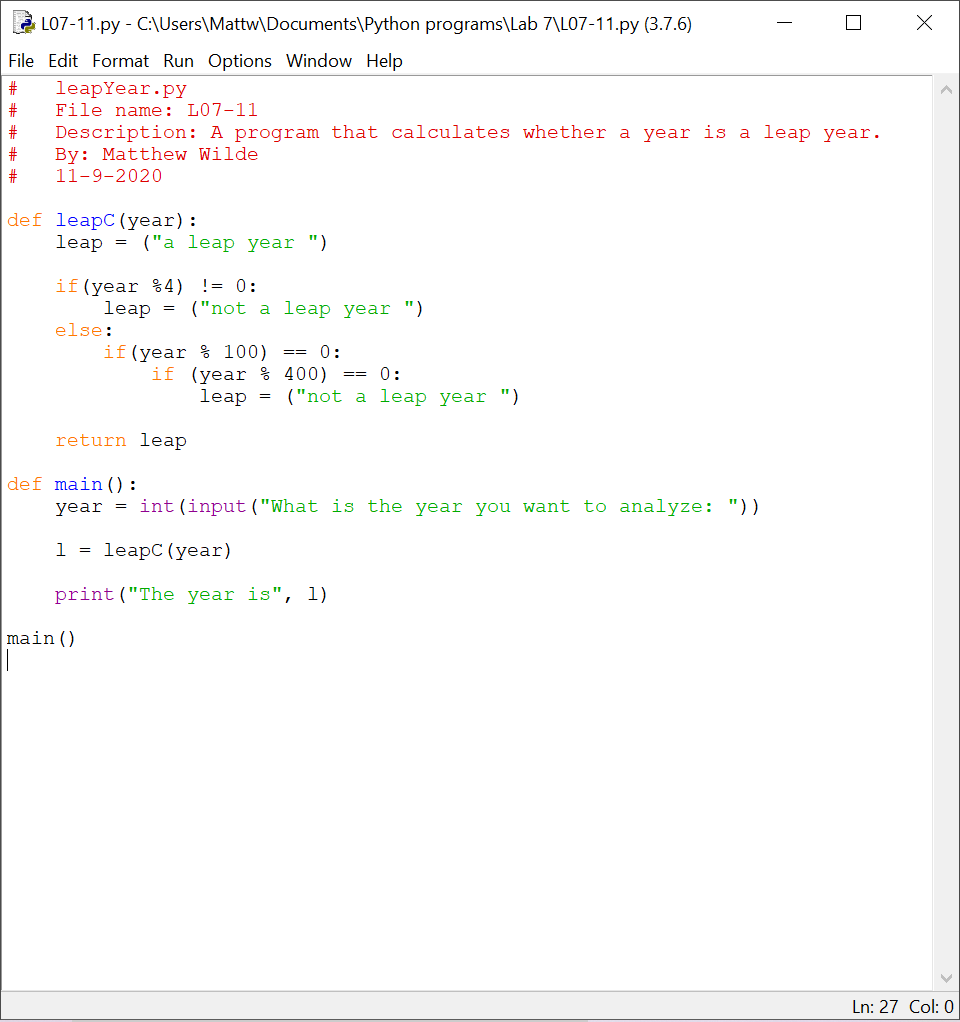
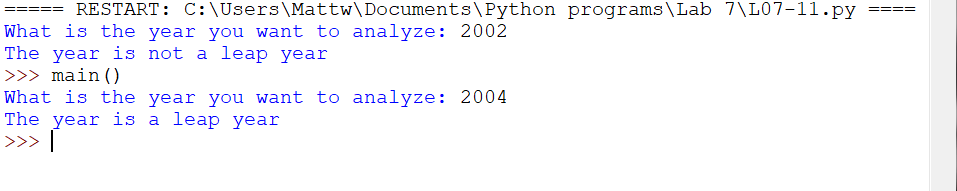
**Written Code:**# easter.py  
# File name: L07-09  
# Description: A program that  
# By: Matthew Wilde  
# 11-9-2020  
  
def easter(year):  
 a = year%19  
 b = year%4  
 c = year%7  
 d = ((19 \* a) + 24)%30  
 e = ((2 \* b)+(4 \* c)+(6 \* d) + 5)%7  
  
 ans = d + e  
   
 return ans  
  
  
   
   
   
  
  
def main():  
 year = int(input("What is the year betweeen 1982 and 2048: "))  
 if year < 1982 or year > 2048:  
 print("Out of range please select a year betweeen 1982 and 2048")  
 year = int(input("What is the year: "))  
  
 e = easter(year)  
  
 print("The date of easter is:",e)  
   
main()

**Screenshots of Code:** **Screenshots of Running Results:**  
**Question CH.07PE:10**  
The formula for Easter in the previous problem works for every year in the range 1900-2099 except for 1954, 1981, 2049, and 2076. For these 4 years it produces a date that is one week too late. Modify the above program to work for the entire range 1900-2099.

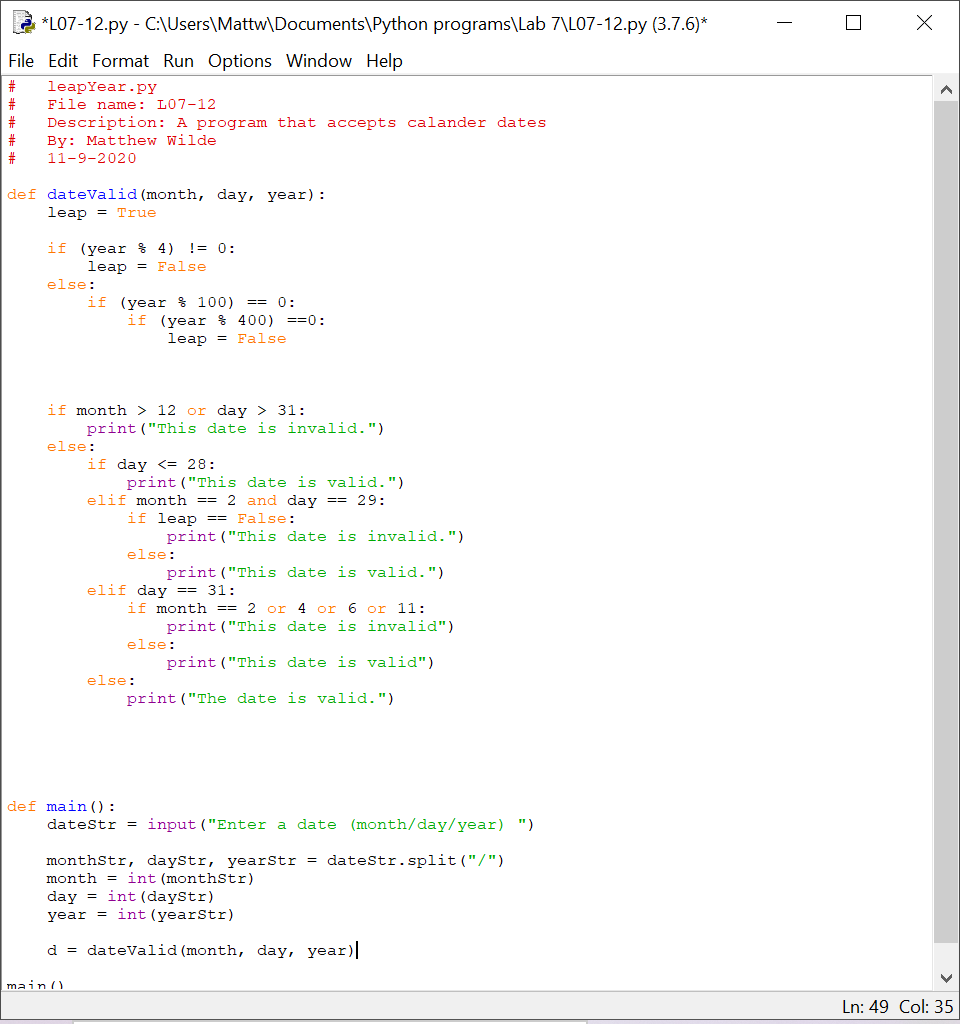
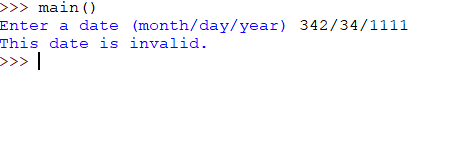
**Written Code:**# easter.py  
# File name: L07-10  
# Description: A program that  
# By: Matthew Wilde  
# 11-9-2020  
  
def easter(year):  
 a = year%19  
 b = year%4  
 c = year%7  
 d = ((19 \* a) + 24)%30  
 e = ((2 \* b)+(4 \* c)+(6 \* d) + 5)%7  
  
 ans = d + e  
   
 return ans  
  
 if year == 1954 or year == 1981 or year == 2049 or year == 2076:  
 return ans - 7  
  
   
   
   
  
  
def main():  
 year = int(input("What is the year betweeen 1982 and 2048: "))  
 if year < 1900 or year > 2099:  
 print("Out of range please select a year betweeen 1982 and 2048")  
 year = int(input("What is the year: "))  
  
 e = easter(year)  
  
 print("The date of easter is:",e)  
   
main()

**Screenshots of Code:** **Screenshots of Running Results:**  
**Question CH.07PE:11**  
A year is a leap year if it is divisible by 4, unless it is a century year that is not divisible by 400. (1800 and 1900 are not leap years while 1600 and 2000 are.) Write a program that calculates whether a year is a leap year.

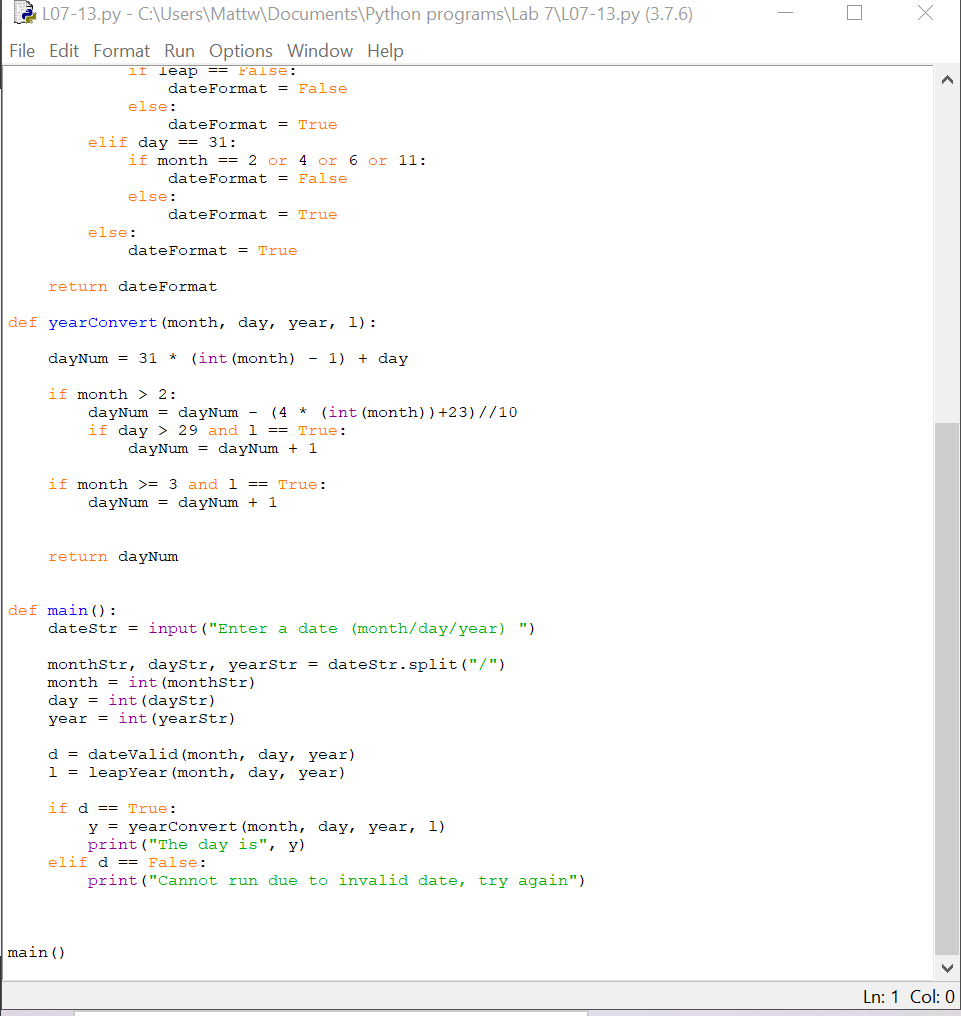
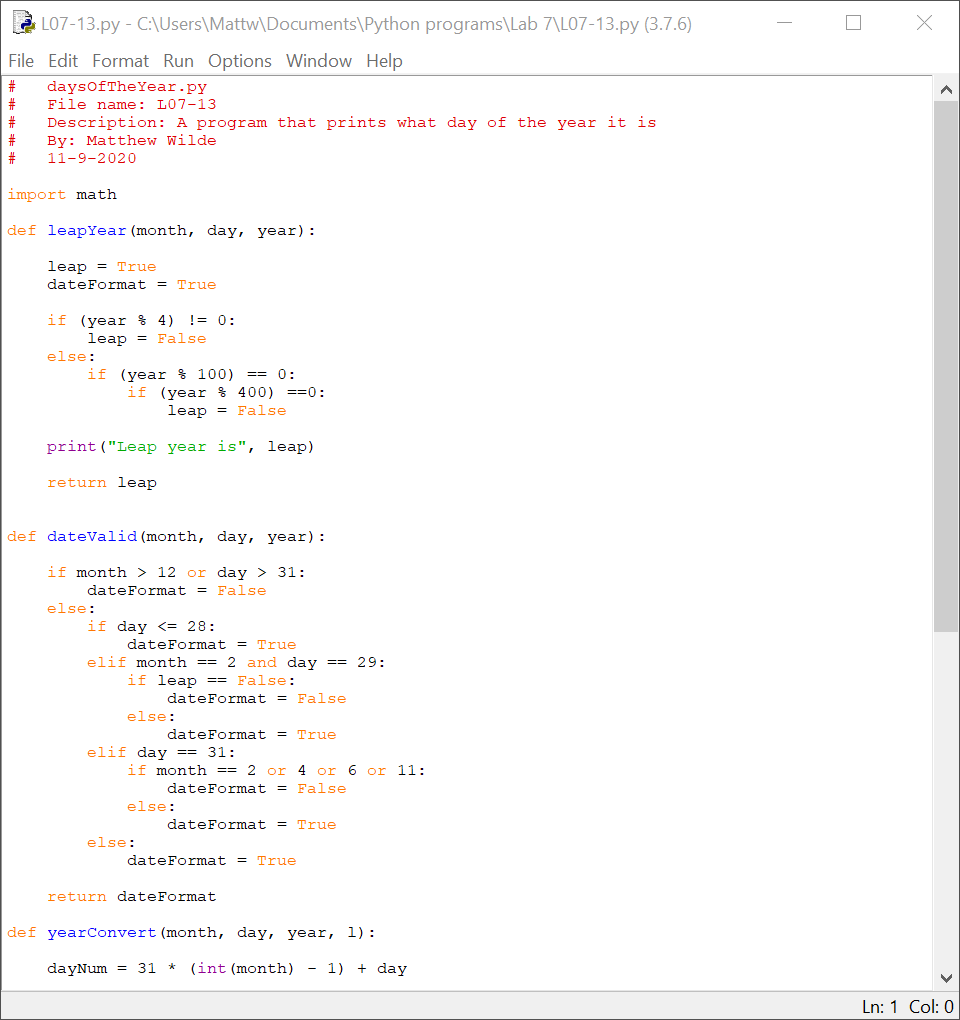
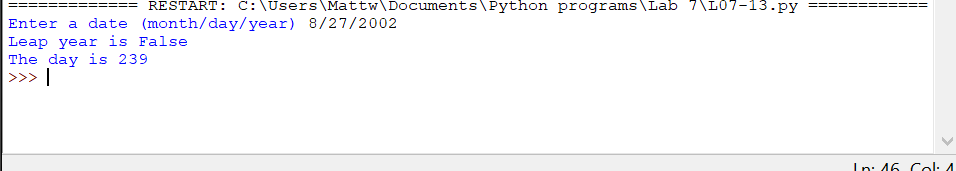
**Written Code:**# leapYear.py  
# File name: L07-11  
# Description: A program that calculates whether a year is a leap year.   
# By: Matthew Wilde  
# 11-9-2020  
  
def leapC(year):  
 leap = ("a leap year ")  
   
 if(year %4) != 0:  
 leap = ("not a leap year ")  
 else:  
 if(year % 100) == 0:  
 if (year % 400) == 0:  
 leap = ("not a leap year ")  
   
 return leap  
  
def main():  
 year = int(input("What is the year you want to analyze: "))  
  
 l = leapC(year)  
  
 print("The year is", l)  
   
main()

**Screenshots of Code:** **Screenshots of Running Results:**  
**Question CH.07PE:12**  
Write a program that accepts a date in the form month! day /year and outputs whether or not the date is valid. For example 5/24/1962 is valid, but 9/31/2000 is not. (September has only 30 days.)

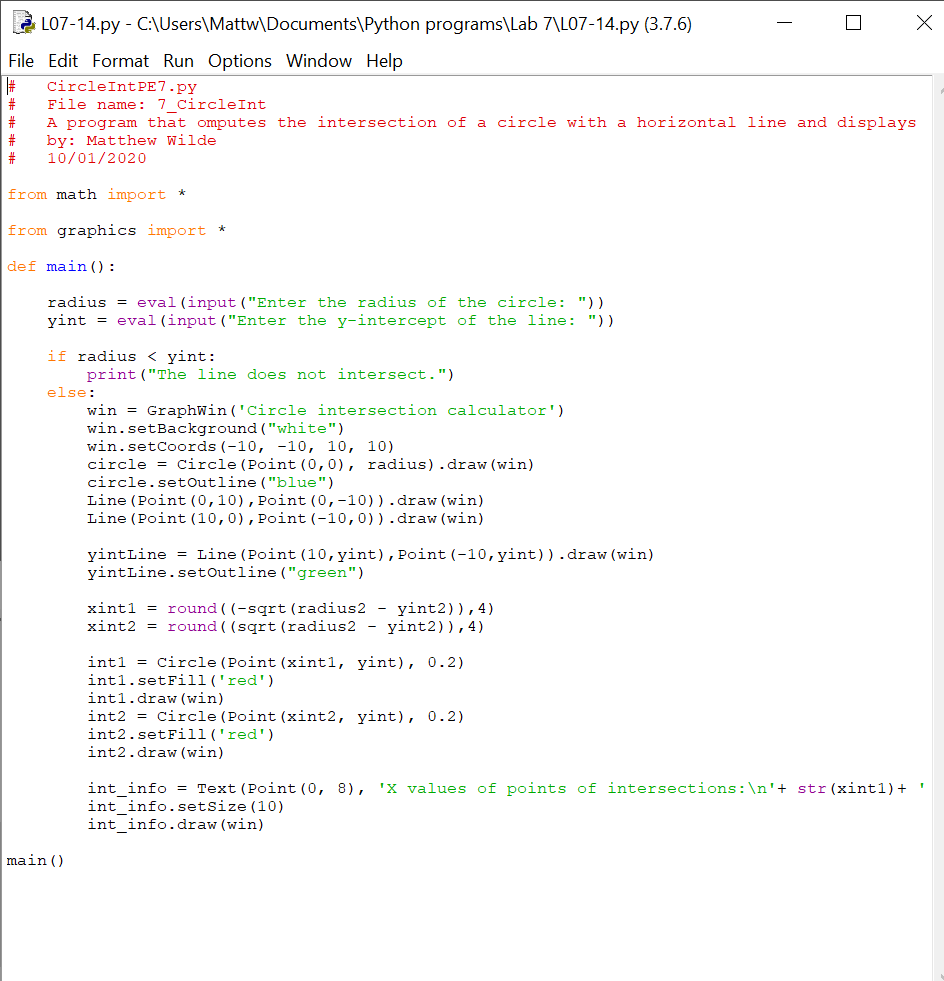
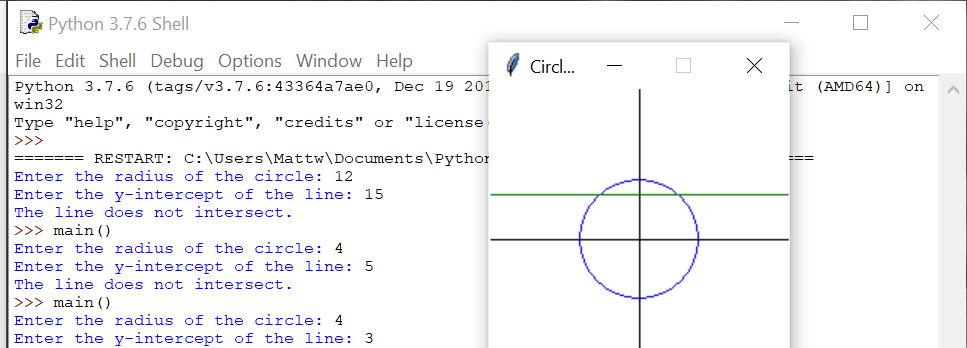
**Written Code:**# leapYear.py  
# File name: L07-12  
# Description: A program that accepts calander dates  
# By: Matthew Wilde  
# 11-9-2020  
  
def dateValid(month, day, year):  
 leap = True  
  
 if (year % 4) != 0:  
 leap = False  
 else:  
 if (year % 100) == 0:  
 if (year % 400) ==0:  
 leap = False  
  
  
  
 if month > 12 or day > 31:  
 print("This date is invalid.")  
 else:  
 if day <= 28:  
 print("This date is valid.")  
 elif month == 2 and day == 29:  
 if leap == False:  
 print("This date is invalid.")  
 else:  
 print("This date is valid.")  
 elif day == 31:  
 if month == 2 or 4 or 6 or 11:  
 print("This date is invalid")  
 else:  
 print("This date is valid")  
 else:  
 print("The date is valid.")  
  
  
  
  
  
def main():  
 dateStr = input("Enter a date (month/day/year) ")  
  
 monthStr, dayStr, yearStr = dateStr.split("/")  
 month = int(monthStr)  
 day = int(dayStr)  
 year = int(yearStr)  
  
 d = dateValid(month, day, year)  
  
main()

**Screenshots of Code:** **Screenshots of Running Results:**  
**Question CH.07PE:13**  
The days of the year are often numbered from 1 through 365 (or 366). This number can be computed in three steps using int arithmetic: Write a program that accepts a date as month/ day /year, verifies that it is a valid date (see previous problem), and then calculates the corresponding day number.

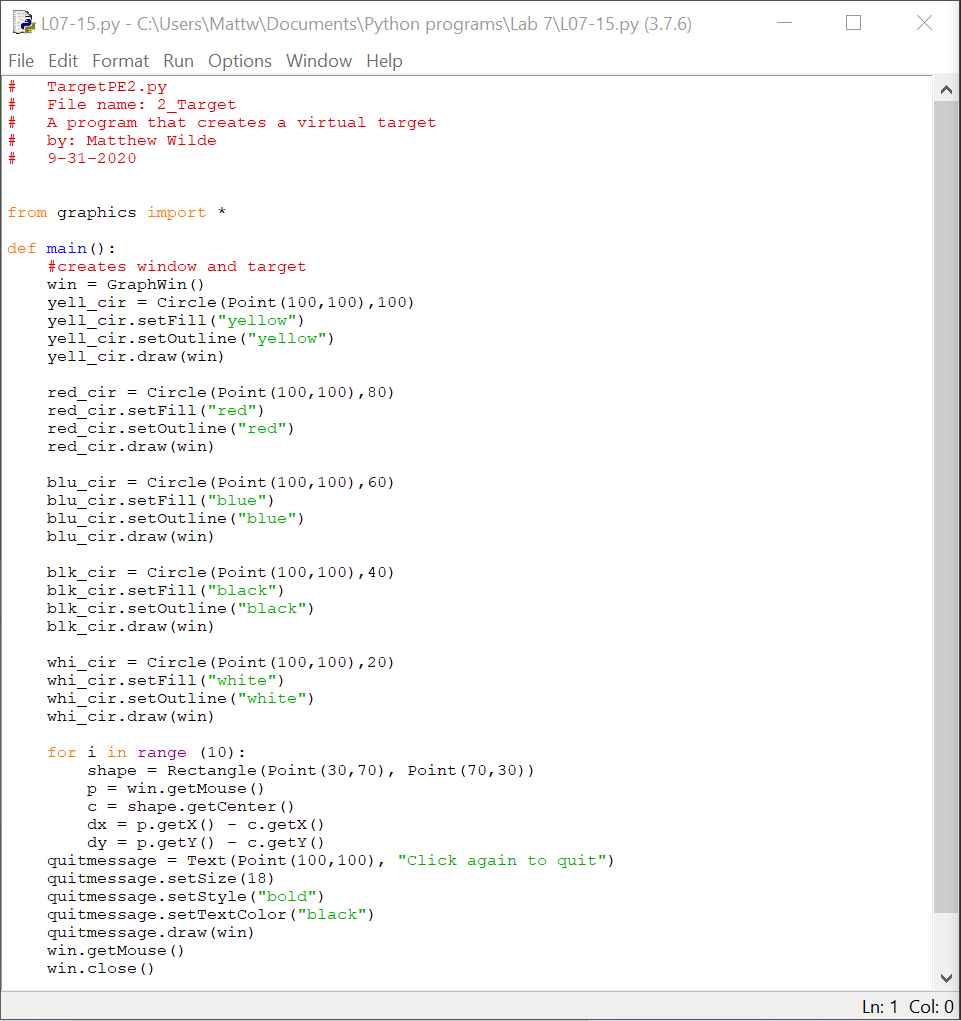
**Written Code:**# daysOfTheYear.py  
# File name: L07-13  
# Description: A program that prints what day of the year it is  
# By: Matthew Wilde  
# 11-9-2020  
  
import math  
  
def leapYear(month, day, year):  
   
 leap = True  
 dateFormat = True  
  
 if (year % 4) != 0:  
 leap = False  
 else:  
 if (year % 100) == 0:  
 if (year % 400) ==0:  
 leap = False  
   
 print("Leap year is", leap)  
  
 return leap  
  
  
def dateValid(month, day, year):  
   
 if month > 12 or day > 31:  
 dateFormat = False  
 else:  
 if day <= 28:  
 dateFormat = True  
 elif month == 2 and day == 29:  
 if leap == False:  
 dateFormat = False  
 else:  
 dateFormat = True  
 elif day == 31:  
 if month == 2 or 4 or 6 or 11:  
 dateFormat = False  
 else:  
 dateFormat = True  
 else:  
 dateFormat = True  
   
 return dateFormat  
  
def yearConvert(month, day, year, l):  
  
 dayNum = 31 \* (int(month) - 1) + day   
  
 if month > 2:  
 dayNum = dayNum - (4 \* (int(month))+23)//10  
 if day > 29 and l == True:  
 dayNum = dayNum + 1  
   
 if month >= 3 and l == True:  
 dayNum = dayNum + 1  
   
   
 return dayNum  
   
  
def main():  
 dateStr = input("Enter a date (month/day/year) ")  
  
 monthStr, dayStr, yearStr = dateStr.split("/")  
 month = int(monthStr)  
 day = int(dayStr)  
 year = int(yearStr)  
  
 d = dateValid(month, day, year)  
 l = leapYear(month, day, year)  
  
 if d == True:  
 y = yearConvert(month, day, year, l)  
 print("The day is", y)  
 elif d == False:  
 print("Cannot run due to invalid date, try again")  
   
   
  
main()

**Screenshots of Code:** **Screenshots of Running Results:**  
**Question CH.07PE:14**  
Do Programming Exercise 7 from Chapter 4, but add a decision to handle the case where the line does not intersect the circle.

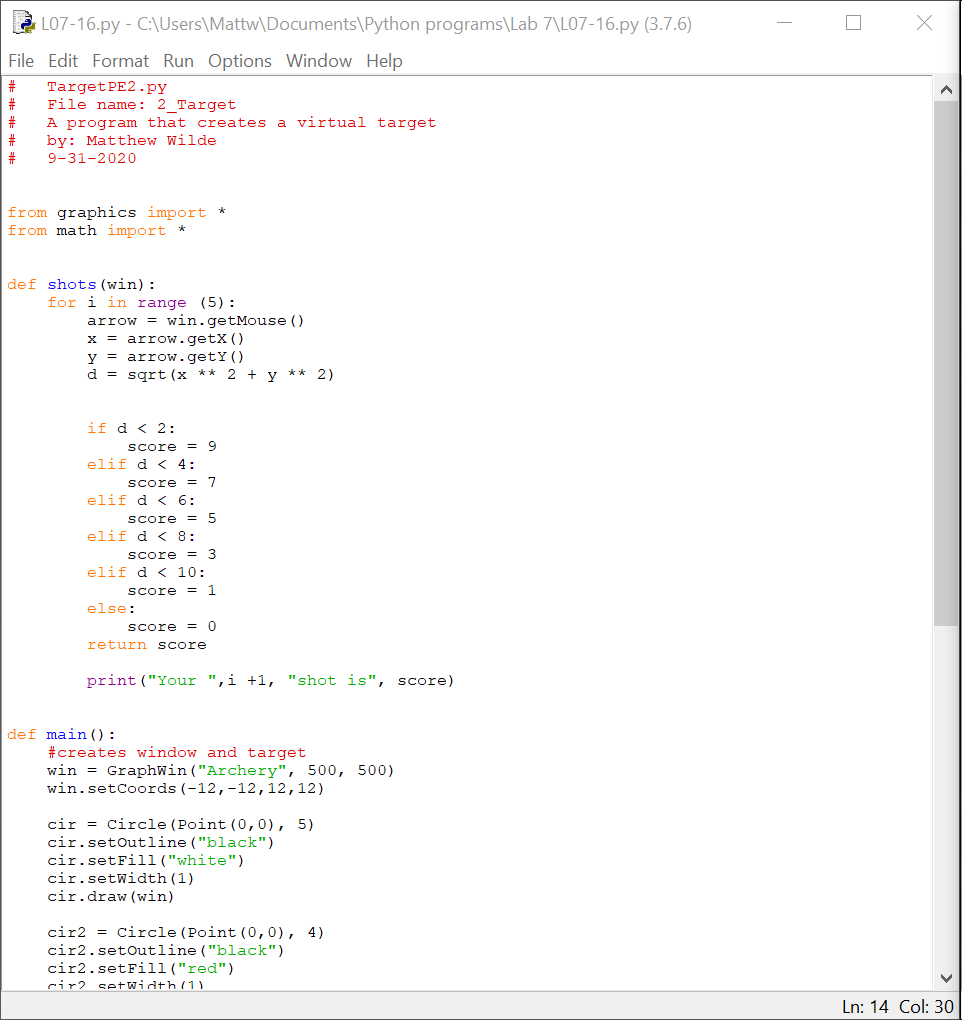
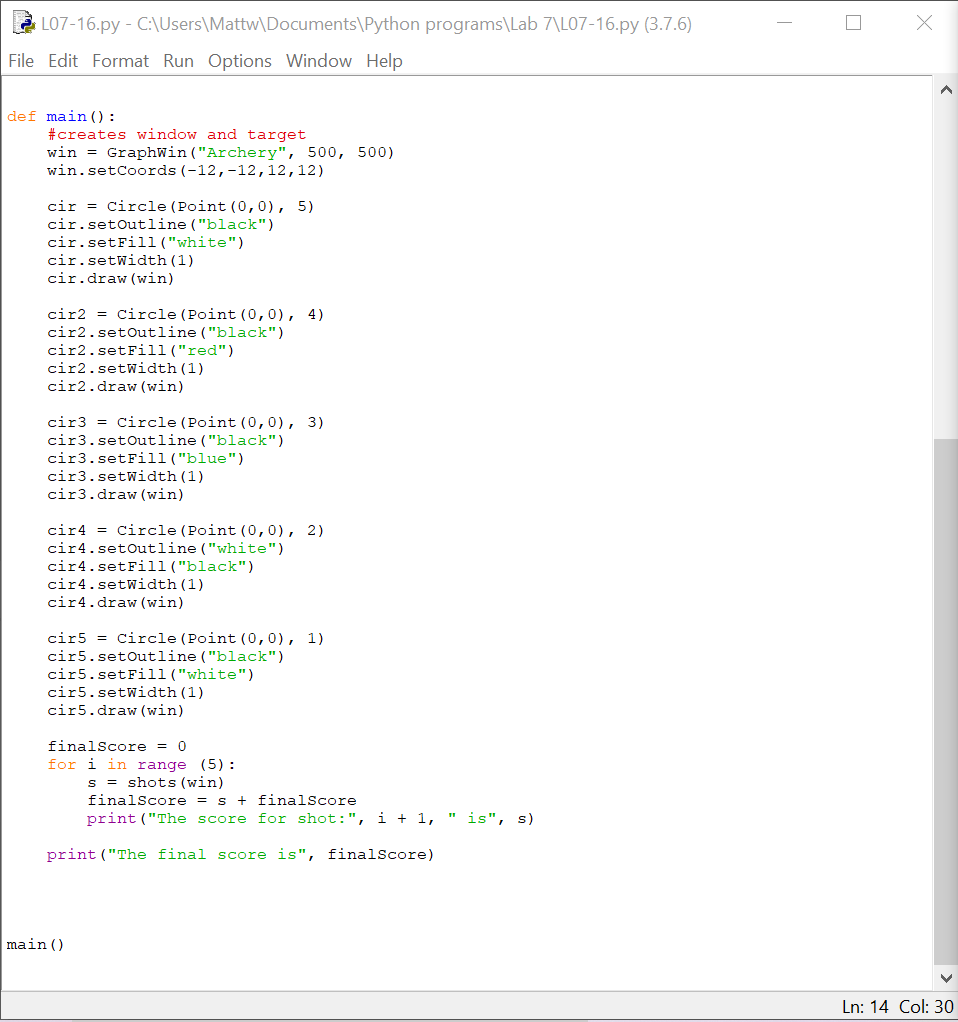
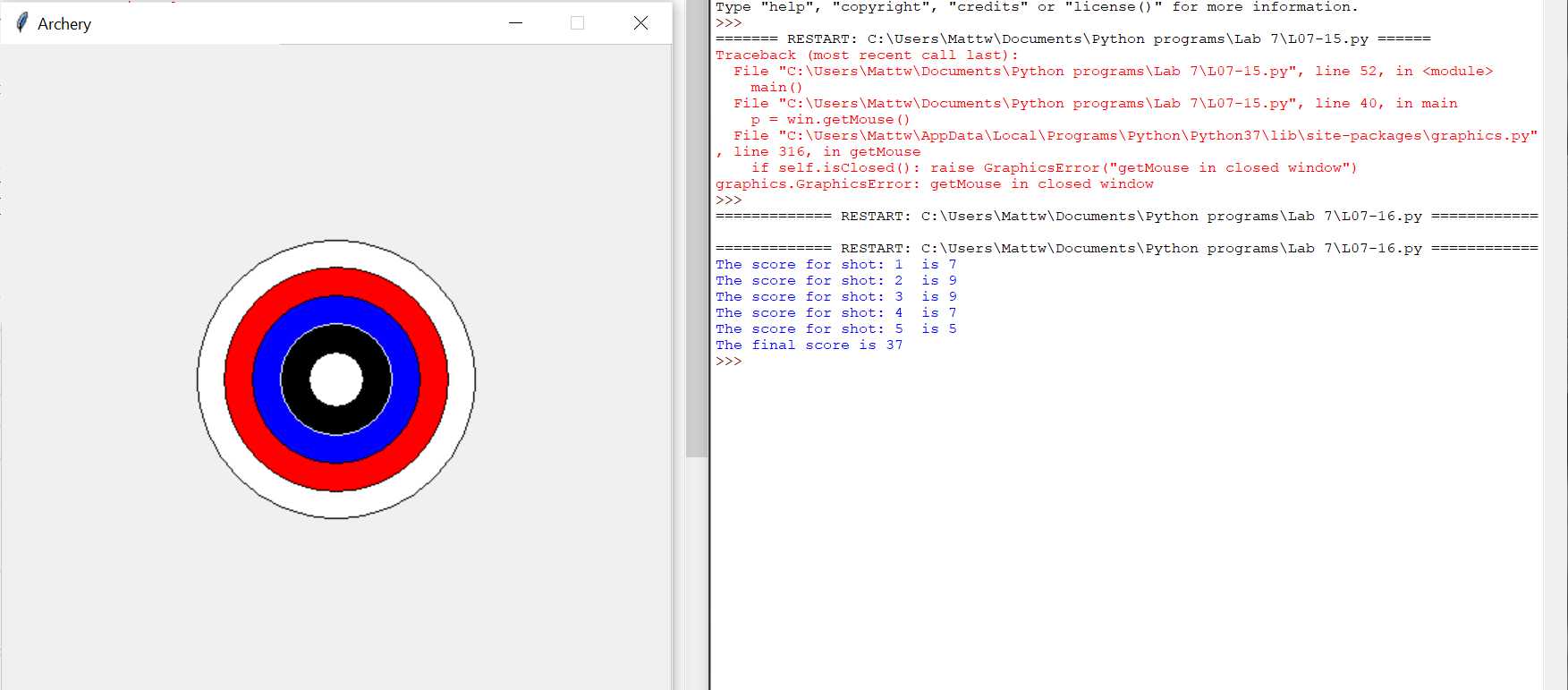
**Written Code:**# CircleIntPE7.py  
# File name: 7\_CircleInt  
# A program that omputes the intersection of a circle with a horizontal line and displays the information textually and graphically.  
# by: Matthew Wilde  
# 10/01/2020  
  
from math import \*   
  
from graphics import \*  
  
def main():  
  
 radius = eval(input("Enter the radius of the circle: "))  
 yint = eval(input("Enter the y-intercept of the line: "))  
  
 if radius < yint:  
 print("The line does not intersect.")  
 else:  
 win = GraphWin('Circle intersection calculator')  
 win.setBackground("white")  
 win.setCoords(-10, -10, 10, 10)  
 circle = Circle(Point(0,0), radius).draw(win)  
 circle.setOutline("blue")  
 Line(Point(0,10),Point(0,-10)).draw(win)  
 Line(Point(10,0),Point(-10,0)).draw(win)  
  
 yintLine = Line(Point(10,yint),Point(-10,yint)).draw(win)  
 yintLine.setOutline("green")  
  
 xint1 = round((-sqrt(radius2 - yint2)),4)  
 xint2 = round((sqrt(radius2 - yint2)),4)  
  
 int1 = Circle(Point(xint1, yint), 0.2)  
 int1.setFill('red')  
 int1.draw(win)  
 int2 = Circle(Point(xint2, yint), 0.2)  
 int2.setFill('red')  
 int2.draw(win)  
  
 int\_info = Text(Point(0, 8), 'X values of points of intersections:\n'+ str(xint1)+ ', ' + str(xint2))  
 int\_info.setSize(10)  
 int\_info.draw(win)  
  
main()

**Screenshots of Code:** **Screenshots of Running Results:**  
**Question CH.07PE:15**  
Do Programming Exercise 8 from Chapter 4, but add a decision to prevent the program from dividing by zero if the line is vertical

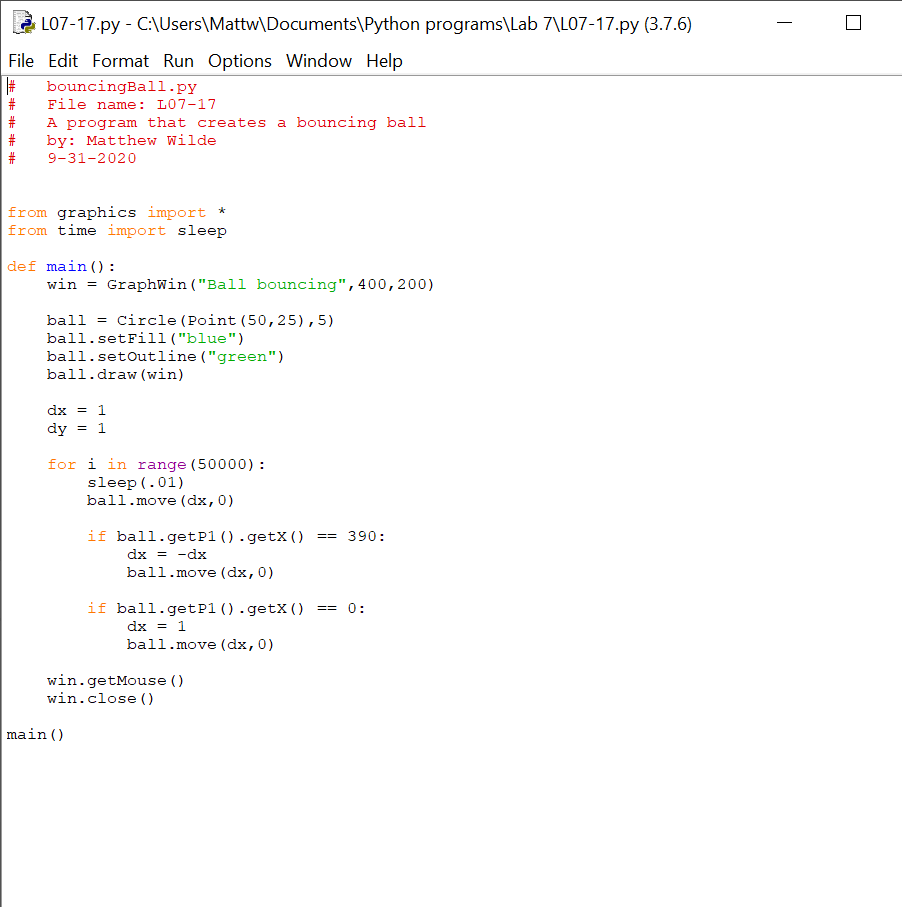
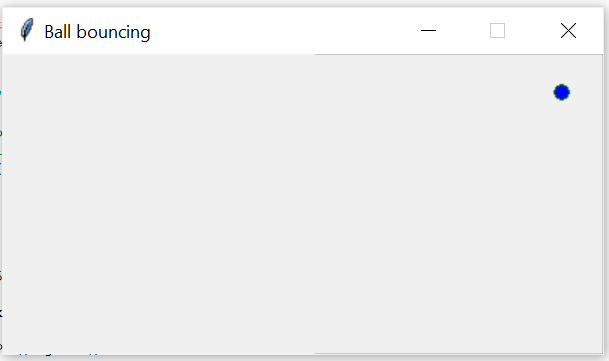
**Written Code:**# TargetPE2.py  
# File name: 2\_Target  
# A program that creates a virtual target  
# by: Matthew Wilde  
# 9-31-2020  
  
  
from graphics import \*  
  
def main():  
 #creates window and target  
 win = GraphWin()  
 yell\_cir = Circle(Point(100,100),100)  
 yell\_cir.setFill("yellow")  
 yell\_cir.setOutline("yellow")  
 yell\_cir.draw(win)  
  
 red\_cir = Circle(Point(100,100),80)  
 red\_cir.setFill("red")  
 red\_cir.setOutline("red")  
 red\_cir.draw(win)  
  
 blu\_cir = Circle(Point(100,100),60)  
 blu\_cir.setFill("blue")  
 blu\_cir.setOutline("blue")  
 blu\_cir.draw(win)  
  
 blk\_cir = Circle(Point(100,100),40)  
 blk\_cir.setFill("black")  
 blk\_cir.setOutline("black")  
 blk\_cir.draw(win)  
  
 whi\_cir = Circle(Point(100,100),20)  
 whi\_cir.setFill("white")  
 whi\_cir.setOutline("white")  
 whi\_cir.draw(win)  
  
 for i in range (10):  
 shape = Rectangle(Point(30,70), Point(70,30))  
 p = win.getMouse()  
 c = shape.getCenter()  
 dx = p.getX() - c.getX()  
 dy = p.getY() - c.getY()  
 quitmessage = Text(Point(100,100), "Click again to quit")  
 quitmessage.setSize(18)  
 quitmessage.setStyle("bold")  
 quitmessage.setTextColor("black")  
 quitmessage.draw(win)  
 win.getMouse()  
 win.close()  
   
main()

**Screenshots of Code:** **Screenshots of Running Results:**  
**Question CH.07PE:16**  
Archery Scorer. Write a program that draws an archery target (see Programming Exercise 2 from Chapter 4) and allows the user to click five times to represent arrows shot at the target. Using five-band scoring, a bulls-eye (yellow) is worth 9 points and each successive ring is worth 2 fewer points down to 1 for white. The program should output a score for each click and keep track of a running sum for the entire series.

**Written Code:**# TargetPE2.py  
# File name: 2\_Target  
# A program that creates a virtual target  
# by: Matthew Wilde  
# 9-31-2020  
  
  
from graphics import \*  
from math import \*  
  
  
def shots(win):  
 for i in range (5):  
 arrow = win.getMouse()  
 x = arrow.getX()  
 y = arrow.getY()  
 d = sqrt(x \*\* 2 + y \*\* 2)  
  
   
 if d < 2:  
 score = 9  
 elif d < 4:  
 score = 7  
 elif d < 6:  
 score = 5  
 elif d < 8:  
 score = 3  
 elif d < 10:  
 score = 1  
 else:  
 score = 0  
 return score  
   
 print("Your ",i +1, "shot is", score)  
  
  
def main():  
 #creates window and target  
 win = GraphWin("Archery", 500, 500)  
 win.setCoords(-12,-12,12,12)  
  
 cir = Circle(Point(0,0), 5)  
 cir.setOutline("black")  
 cir.setFill("white")  
 cir.setWidth(1)  
 cir.draw(win)  
  
 cir2 = Circle(Point(0,0), 4)  
 cir2.setOutline("black")  
 cir2.setFill("red")  
 cir2.setWidth(1)  
 cir2.draw(win)  
  
 cir3 = Circle(Point(0,0), 3)  
 cir3.setOutline("black")  
 cir3.setFill("blue")  
 cir3.setWidth(1)  
 cir3.draw(win)  
  
 cir4 = Circle(Point(0,0), 2)  
 cir4.setOutline("white")  
 cir4.setFill("black")  
 cir4.setWidth(1)  
 cir4.draw(win)  
  
 cir5 = Circle(Point(0,0), 1)  
 cir5.setOutline("black")  
 cir5.setFill("white")  
 cir5.setWidth(1)  
 cir5.draw(win)  
  
 finalScore = 0   
 for i in range (5):  
 s = shots(win)  
 finalScore = s + finalScore  
 print("The score for shot:", i + 1, " is", s)  
  
 print("The final score is", finalScore)  
  
   
   
   
main()

**Screenshots of Code:** **Screenshots of Running Results:**  
**Question CH.07PE:17**  
Write a program to animate a circle bouncing around a window. The basic idea is to start the circle somewhere in the interior of the window. Use variables dx and dy (both initialized to 1) to control the movement of the circle. Use a large counted loop (say 10000 iterations), and each time through the loop move the circle using dx and dy . When the x-value of the center of the circle gets too high (it hits the edge), change dx to -1. When it gets too low, change dx back to 1. Use a similar approach for dy .

**Written Code:**# bouncingBall.py  
# File name: L07-17  
# A program that creates a bouncing ball  
# by: Matthew Wilde  
# 9-31-2020  
  
  
from graphics import \*  
from time import sleep  
  
def main():  
 win = GraphWin("Ball bouncing",400,200)  
  
 ball = Circle(Point(50,25),5)  
 ball.setFill("blue")  
 ball.setOutline("green")  
 ball.draw(win)  
  
 dx = 1  
 dy = 1  
  
 for i in range(50000):  
 sleep(.01)  
 ball.move(dx,0)  
   
 if ball.getP1().getX() == 390:  
 dx = -dx  
 ball.move(dx,0)  
   
 if ball.getP1().getX() == 0:  
 dx = 1  
 ball.move(dx,0)  
   
 win.getMouse()  
 win.close()  
   
main()

**Screenshots of Code:** **Screenshots of Running Results:**  
**Question CH.07PE:18**  
Take a favorite programming problem from a previous chapter and add decisions and/ or exception handling as required to make it truly robust (will not crash on any inputs). Trade your program with a friend and have a contest to see who can ''break" the other's program.

**Written Code:**# grade.py  
# File name: L07-18  
# A program that computes a students grade  
# by: Matthew Wilde  
# 10-29-2020  
  
import math  
  
def grade(score):  
   
 if score > 100:  
 return("That is not a valid score ")  
 if score < 0:  
 return("That is not a valid score ")  
 if score >= 90 or score == 100:  
 return("A")  
 elif score >= 80 :  
 return("B")  
 elif score >= 70:  
 return("C")  
 elif score >= 60:  
 return("D")  
 elif score < 60:  
 return("F")  
   
  
def main():  
   
 quizGrade = eval(input("What did you get on your quiz (number) "))  
  
 g = grade(quizGrade)  
  
 print("Your grade is:", g)  
  
  
main()

**Screenshots of Code:** **Screenshots of Running Results:**