

## Homework 1

Michael Quach

### Problem 1

#Michael Quach

#Problem 1. Salary Computation

#Write a Python program called p1.py that computes the salary for a worker that is paid weekly.

#The p1 program starts by displaying some instructions to the user.

#Then it reads from the terminal (in this order!) the number of hours worked in a week (variable

#hours\_worked, to convert to float), the hourly salary (rate\_per\_hour, float), and whether the

#worker has received a bonus (variable yes\_no). If yes\_no is 'y', then the program reads the

#bonus (in variable bonus, also converted to a float).

#The program computes the total salary as the sum of the overtime salary, the non-overtime

#salary, and the bonus. At the end, the program displays the total salary and the overtime pay.

#Any hours worked in a week above 40 will be paid with a salary rate that is 1.5 times the regular hourly rate.

```
print ("This program determines the weekly salary for an employee.")
```

```
print ("Salary = hours*rate + overtime + bonus")
```

```
print ("Overtime is 1.5*rate for any hours beyond 40.")
```

```
hours=input("Enter the number of hours worked this week: ")
```

```
hours=float(hours)
```

```
rates=input("Enter the salary rate per hour (do not include the '$' sign): ")
```

```
rate=float(rates)
```

```
bonus=0
```

```
#get bonus amount, if any
```

```
bonustr=input("Did the worker get a bonus ? (y/n) ")
```

```
if bonustr=='y':
```

```
    bonus=input("How much was the bonus?")
```

```
    bonus=float(bonus)
```

```
#Calculation if there was overtime
```

```
if hours>40:
```

```
    overtime=(hours-40)*(rate*1.5)
```

```
    print("The worker has earned $", overtime+(40*rate)+bonus, "($", overtime, " overtime).")
```

```
#Calculation if there was no overtime
```

```
elif hours<=40:
```

```
    print("The worker has earned $", (hours*rate)+bonus, " with no overtime.")
```

```

p1.py - C:\Users\Beefdi\Desktop\p1.py (3.6.2)
File Edit Format Run Options Window Help

#Michael Quach

#Problem 1. Salary Computation
#Write a Python program called p1.py that computes the salary for a worker that
#The p1 program starts by displaying some instructions to the user.
#Then it reads from the terminal (in this order!) the number of hours worked in
#hours_worked, to convert to float), the hourly salary (rate_per_hour, float), a
#worker has received a bonus (variable yes_no). If yes_no is 'y', then the progra
#bonus (in variable bonus, also converted to a float).
#The program computes the total salary as the sum of the overtime salary, the no
#salary, and the bonus. At the end, the program displays the total salary and th
#Any hours worked in a week above 40 will be paid with a salary rate that is 1.5

print ("This program determines the weekly salary for an employee.")
print ("Salary = hours*rate + overtime + bonus")
print ("Overtime is 1.5*rate for any hours beyond 40.")
hours=input("Enter the number of hours worked this week: ")
hours=float(hours)
rates=input("Enter the salary rate per hour (do not include the '$' sign): ")
rate=float(rates)
bonus=0
#get bonus amount, if any
bonustr=input("Did the worker get a bonus ? (y/n) ")
if bonustr=='y':
    bonus=input("How much was the bonus?")
    bonus=float(bonus)
#Calculation if there was overtime
if hours>40:
    overtime=(hours-40)*(rate*1.5)
    print("The worker has earned $", overtime+(40*rate)+bonus, "($",overtime," o
#Calculation if there was no overtime
elif hours<=40:
    print("The worker has earned $", (hours*rate)+bonus, " with no overtime.")

Python 3.6.2 Shell
Python 3.6.2 (v3.6.2:5fd33b5, Jul 8 2017, 04:14:34) [MSC v.1900 32 bit (Intel)]
on win32
Type "copyright", "credits" or "license()" for more information.
>>>
===== RESTART: C:\Users\Beefdi\Desktop\p1.py =====
This program determines the weekly salary for an employee.
Salary = hours*rate + overtime + bonus
Overtime is 1.5*rate for any hours beyond 40.
Enter the number of hours worked this week: 50
Enter the salary rate per hour (do not include the '$' sign): 13
Did the worker get a bonus ? (y/n) n
The worker has earned $ 715.0 ($ 195.0 overtime).
>>>
===== RESTART: C:\Users\Beefdi\Desktop\p1.py =====
This program determines the weekly salary for an employee.
Salary = hours*rate + overtime + bonus
Overtime is 1.5*rate for any hours beyond 40.
Enter the number of hours worked this week: 30
Enter the salary rate per hour (do not include the '$' sign): 16
Did the worker get a bonus ? (y/n) y
How much was the bonus?5
The worker has earned $ 485.0 with no overtime.
>>>
===== RESTART: C:\Users\Beefdi\Desktop\p1.py =====
This program determines the weekly salary for an employee.
Salary = hours*rate + overtime + bonus
Overtime is 1.5*rate for any hours beyond 40.
Enter the number of hours worked this week: 100
Enter the salary rate per hour (do not include the '$' sign): 100
Did the worker get a bonus ? (y/n) y
How much was the bonus?100
The worker has earned $ 13100.0 ($ 9000.0 overtime).
>>>

```

## Problem 2

#Michael Quach

```
#####
##
```

#Write all code for this problem in a file p2.py.

#1. The program consists of a main while loop (an infinite loop) in which the user is prompted to enter values for coefficients a, b, and c.

#Assume the user types valid float numbers from the terminal.

#The program converts the input to float type and then uses formula (1) above to compute solutions x1 and x2, as follows:

#a) if  $b^2-4ac < 0$  then the solutions are complex numbers (i.e. not real) and the program displays the string "no real solutions",

#b) if  $b^2-4ac = 0$  then  $x_1 = x_2$  and the program displays: "one solution: " followed by the value x1.

#c) if  $b^2-4ac > 0$  then the solutions are distinct and the program displays "two solutions: "

#followed by the values of x1 and x2.

#To keep the problem simple we can assume that the user never enters a value for coefficient a that is equal to 0.

#To stop the loop and to end the program, the user types the enf-of-file key – CTRL-Z on Windows

#(CTRL-D on Linux/Mac/Unix) – when expected to enter coefficient a for a new iteration.

```
#####
##
```

**import** pylab

**import** math

#NOTICE: Since you had us install Python via Anaconda, I couldn't figure out

#how to install the pylab module for IDLE, so I'm using Spyder from the  
#Anaconda package because it can access pylab.

coA = 0

**print**("To exit the program, enter 0 when asked for coefficient a.")

#I couldn't figure out how to use EOF as a condition, and every answer online just says to use  
some method or other.

**while** 1:

coA=input("Please enter the value of the coefficient a: ")

coA=float(coA)

**if** coA == 0: **break**

coB=input("Please enter the value of the coefficient b: ")

coB=float(coB)

coC=input("Please enter the value of the coefficient c: ")

coC=float(coC)

coefficient=(coB\*coB)-(4\*coA\*coC)

**if** coefficient<0:

**print**("No real solutions.")

**elif** coefficient==0:

x1=-coB/(2\*coA)

**print**("One solution: ",x1)

**elif** coefficient>0:

x1=(-coB+math.sqrt(coefficient))/(2\*coA)

x2=(-coB-math.sqrt(coefficient))/(2\*coA)

**print**("Two solutions:", x1," and ",x2)

#####

##

#2. Within the main loop, after displaying the values of the real solutions (if any) the program  
must display the graphic of the quadratic function on the [-5, 5] domain.

#For that, use the pylab Python module. Within a nested while loop populate a list of floats in  
variable xs with 100 float numbers

#between -5 and +5 as shown in the lecture and append to a list ys the matching function  
values:

# for each x in xs,

# append in ys the value of expression  $a*x*x + b*x + c$ .

#(As an alternative to a loop, you could use the pylab.linspace() function to generate the x  
 values.)

#Then, use the pylab.plot() function to generate the plot and the pylab.show() function to display  
the graphic figure.

#Make sure the function line is displayed with a blue line and dots.

#####

##

```

xs=[]
ys=[]
x0=-5
x1=+5
x=x0
n=100
dx=(x1-x0)/n

while x<=x1:
    xs.append(x)
    y=(coA*x*x)+(coB*x)+coC
    ys.append(y)
    x+=dx

pylab.plot(xs, ys, "b.-")
pylab.show()

```

The screenshot shows the Spyder Python IDE with a file named `temp.py` open. The code in the editor is a quadratic equation solver that prompts the user for coefficients `a`, `b`, and `c`. It then calculates the discriminant and prints the real solutions if they exist. The program also generates a plot of the quadratic function  $y = ax^2 + bx + c$  using `pylab`. The plot shows a parabola opening upwards with its vertex at  $(0, -10)$  and x-intercepts at  $x = -5$  and  $x = 5$ . The x-axis ranges from -5 to 5, and the y-axis ranges from -20 to 120. The plot is displayed in the Python console window.

```

16 import pylab
17 import math
18 #NOTICE: Since you had us install Python via Anaconda, I couldn't figure out
19 #how to install the pylab module for IDLE, so I'm using Spyder from the
20 #Anaconda package because it can access pylab.
21 coA = 0
22 print("To exit the program, enter 0 when asked for coefficient a.")
23 #I couldn't figure out how to use EOF as a condition, and every answer online
24 #just says to use some method or other.
25 while 1:
26     coA=input("Please enter the value of the coefficient a: ")
27     coA=float(coA)
28     if coA == 0: break
29     coB=input("Please enter the value of the coefficient b: ")
30     coB=float(coB)
31     coC=input("Please enter the value of the coefficient c: ")
32     coC=float(coC)
33     coefficient=(coB*coB)-(4*coA*coC)
34     if coefficient<0:
35         print("No real solutions.")
36     elif coefficient==0:
37         x1=-coB/(2*coA)
38         print("One solution: ",x1)
39     elif coefficient>0:
40         x1=(-coB-math.sqrt(coefficient))/(2*coA)
41         x2=(-coB+math.sqrt(coefficient))/(2*coA)
42         print("Two solutions: ", x1, " and ",x2)
43
44 #####
45 #2. Within the main loop, after displaying the values of the real solutions (if any) the program must display the
46 #for that, use the pylab Python module. Within a nested while loop populate a list of floats in variable xs with
47 #between -5 and +5 as shown in the Lecture and append to a list ys the matching function values:
48 # for each x in xs,
49 # append in ys the value of expression a*x*x + b * x + c.
50 #(As an alternative to a loop, you could use the pylab.linspace() function to generate the x values.)
51 #Then, use the pylab.plot() function to generate the plot and the pylab.show() function to display the graphic fi
52 #Make sure the function line is displayed with a blue line and dots.
53 #####
54
55 xs=[]
56 ys=[]
57 x0=-5

```

## Problem 3

Algorithm:

```

input=user_input;
number_of_quarters = input/0.25 [without remainder]
input -= number_of_quarters*0.25
number_of_dimes = input
Input -= number_of_dimes*0.10
Number_of_pennies = input*100

```

```

#Michael Quach
#####
##
#Problem 3. Computing Change (includes 10 extra credit points)
#Write a program p3.py that computes the equivalent of a dollar amount in
#change using quarters, dimes, and pennies. No nickels are used for conversion.
#The program reads from the terminal the dollar amount in a loop while not
#end-of-file. Inside the loop it computes how many quarters, dimes, and pennies
#make up the original dollar amount and then displays the change. The program
#should terminate if the user types an invalid string.
#####
##
while 1:
    changed=input("Enter amount to exchange: ")
    changed=float(changed)
    if type(changed)!=float:
        break
    change=changed
    quarters=change//0.25
    change=change%0.25
    dimes=change//0.10
    change=change%0.10
    pennies=(change*100)
    coins=quarters+dimes+pennies
    print("$", changed, " makes ", quarters, "quarters, ", dimes, "dimes, and "\
        , pennies, "pennies, totalling $",(quarters*0.25)+(dimes*0.10)+\
        (pennies/100), "across", coins, " coins.")

```

```
p3.py - C:\Users\Beefdip\Desktop\p3.py (3.6.2) Python 3.6.2 Shell

File Edit Format Run Options Window Help File Edit Shell Debug Options Window Help

##### RESTART: C:\Users\Beefdip\Desktop\p3.py #####
>>>
Enter amount to exchange: 2.5
$ 2.5 makes 10.0 quarters, 0.0 dimes, and 0.0 pennies, totalling $ 2.5 across 10.0 coins.
Enter amount to exchange: 5
$ 5.0 makes 20.0 quarters, 0.0 dimes, and 0.0 pennies, totalling $ 5.0 across 20.0 coins.
Enter amount to exchange: 5.2
$ 5.2 makes 20.0 quarters, 2.0 dimes, and 1.6653345369377348e-14 pennies, totalling $ 5.2 across 22.0000000000000018 coins.
Enter amount to exchange: 7.09
$ 7.09 makes 28.0 quarters, 0.0 dimes, and 8.9999999999999986 pennies, totalling $ 7.09 across 36.999999999999996 coins.
Enter amount to exchange: 7.75
$ 7.75 makes 31.0 quarters, 0.0 dimes, and 0.0 pennies, totalling $ 7.75 across 31.0 coins.
Enter amount to exchange: 6.65
$ 6.65 makes 26.0 quarters, 1.0 dimes, and 5.0000000000000035 pennies, totalling $ 6.65 across 32.0000000000000036 coins.
Enter amount to exchange: 6.64
$ 6.64 makes 26.0 quarters, 1.0 dimes, and 3.9999999999999976 pennies, totalling $ 6.64 across 30.999999999999998 coins.
Enter amount to exchange: 10
$ 10.0 makes 40.0 quarters, 0.0 dimes, and 0.0 pennies, totalling $ 10.0 across 40.0 coins.
Enter amount to exchange: 7.74
$ 7.74 makes 30.0 quarters, 2.0 dimes, and 4.000000000000002 pennies, totalling $ 7.74 across 36.000000000000002 coins.
Enter amount to exchange: 9.99
$ 9.99 makes 39.0 quarters, 2.0 dimes, and 4.000000000000002 pennies, totalling $ 9.99 across 45.000000000000002 coins.
Enter amount to exchange: 5.00
$ 5.0 makes 20.0 quarters, 0.0 dimes, and 0.0 pennies, totalling $ 5.0 across 20.0 coins.
Enter amount to exchange: 7.89
$ 7.89 makes 31.0 quarters, 1.0 dimes, and 3.9999999999999976 pennies, totalling $ 7.89 across 35.9999999999999964 coins.
Enter amount to exchange:
Traceback (most recent call last):

#####
#Michael Quach
#####
#Problem 3. Computing Change (includes 10 extra credit points)
#Write a program p3.py that computes the equivalent of a dollar amount in change using quarters, dimes, and pennies. No nickels are used for conversion.
#The program reads from the terminal the dollar amount in a loop while not end-of-file. Inside the loop it computes how many quarters, dimes, and pennies make up the original dollar amount and then displays the change. The program should terminate if the user types an invalid string.
#####
while 1:
    changed=input("Enter amount to exchange: ")
    changed=float(changed)
    if type(changed)!=float:
        break
    change=changed
    quarters=change//0.25
    change=change%0.25
    dimes=change//0.10
    change=change%0.10
    pennies=(change*100)
    coins=quarters+dimes+pennies
    print("$", changed, " makes ", quarters, " quarters, ", dimes, " dimes, and ", pennies, " pennies, totalling $", (quarters*0.25)+(dimes*0.10)+(pennies/100), " across", coins, " coins.")

Ln: 21 Col: 12 Ln: 55 Col: 4
```