Solutions for

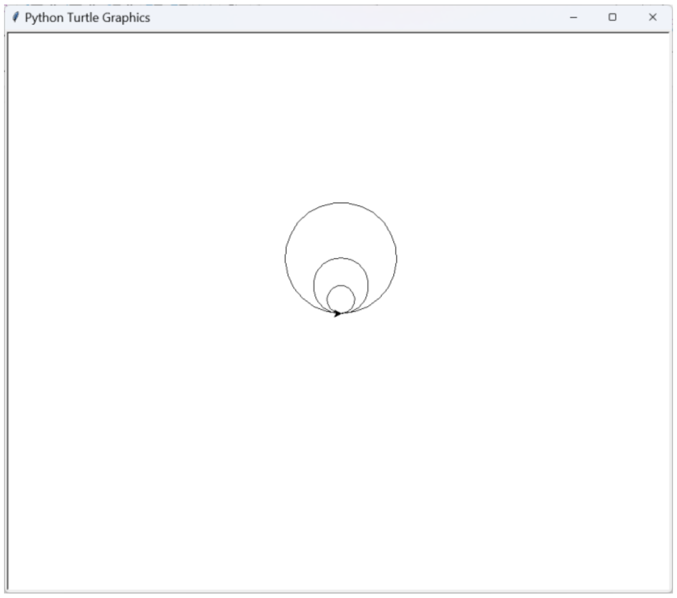
Try A Nybble of Python

Chapter Practices

Chapter 1 Practice

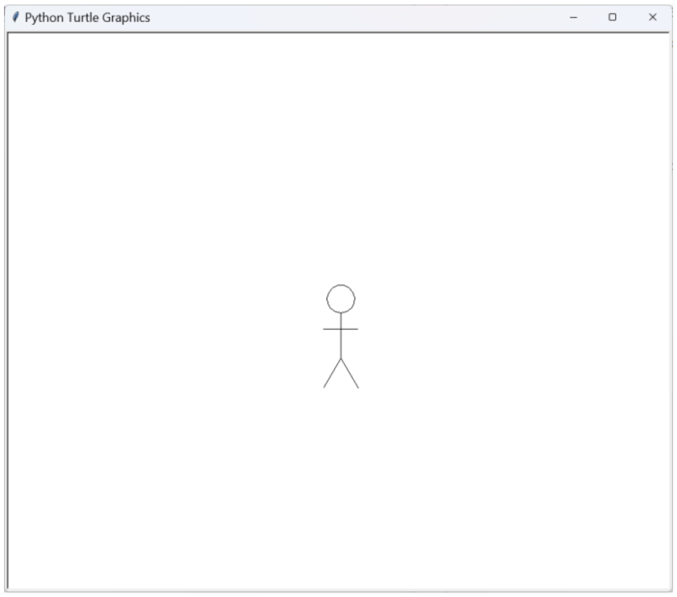
1. Results vary.
2.

```
import turtle
t = turtle.Pen()
t.circle(20)
t.circle(40)
t.circle(80)
```



stickman_practice.py

```
import turtle
t= turtle.Pen()

t.circle(20)    #new head


t.right(90)

t.forward(65)
t.left(30)
t.forward(50)
t.backward(50)
t.right(60)
t.forward(50)

t.penup()
t.right(150)
t.forward(85)
t.right(90)
t.pendown()

t.forward(50)
t.hideturtle()
```



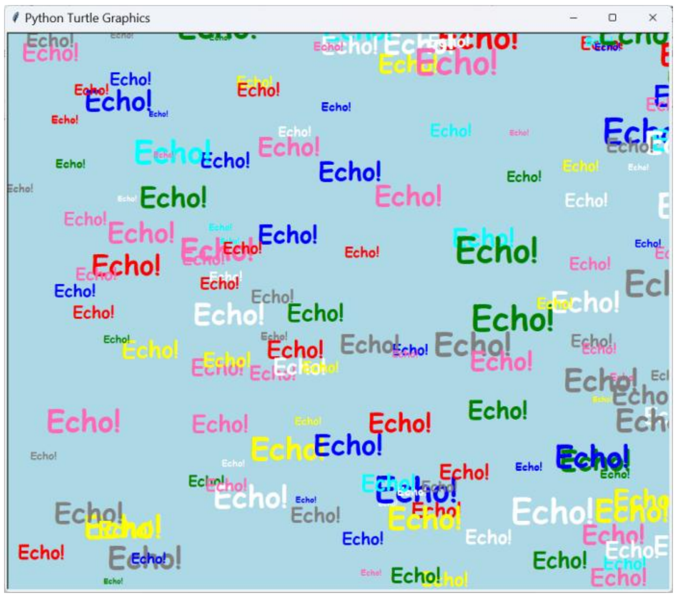
Chapter 2 Practice

1. Finish chapter programs.

2.

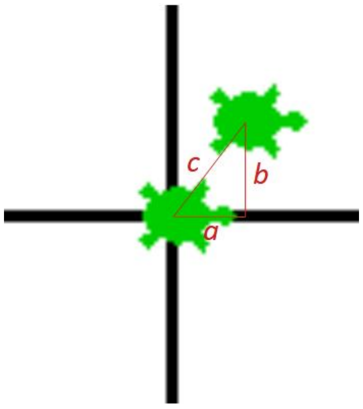
LOL_practice.py

```
import random
import turtle

t = turtle.Pen()

t.hideturtle()
t.speed(0)

turtle.bgcolor("light blue")
colorList = ["white", "red", "green", "blue", "yellow", "aqua", "hot pink", "gray"]

right_side = turtle.window_width()//2
left_side = -right_side
top_side =  turtle.window_height()//2
bottom_side = -top_side

for i in range(150):
    t.pencolor(random.choice(colorList))    #picks a random pen color

    x = random.randint(left_side, right_side)     #picks random x coordinate
    y = random.randint(bottom_side, top_side)     #picks random y coordinate

    t.penup()
    t.setpos(x,y)
    t.pendown()

    font_size = random.randint(8,36)
    t.write("Echo!", font = ("Comic Sans MS", font_size, "bold"))
```



3. Using the information below and the Pythagorean theorem, solve for how many pixels the clicked stamp at (18.00,24.00) is from the center stamp at (0.00,0.00). It might help to sketch a right triangle.

Pythagorean theorem: $a^2 + b^2 = c^2$, where *a* and *b* are legs, and *c*, the hypotenuse, is opposite of the right angle.



a lies along the horizontal x-axis, so the difference between the x coordinates is the distance of a.

a = 18.00 – 0.00
a = 18.00

b lies vertically up and down, so the difference between the y coordinates is the distance of b.

b = 24.00 – 0.00
b = 24.00

Since a and b intersect to form a right angle, the red triangle drawn above is a right triangle! Use the Pythagoream theorem to solve for the distance of c, which is the distance between the two clicks!

Plug the values of a and b into the Pythagorean theorem.

$(18)^2 + (24)^2 = c^2$
$324 + 576 = c^2$
$900 = c^2$
$\sqrt{900} = \sqrt{c^2}$
$30 = c$

The distance between the two clicks is 30 pixels!

Chapter 3 Practice

1. The Pythagorean Theorem, $a^2 + b^2 = c^2$, is used to solve for c.
   Solve for `c` by taking each side of the equation to the half power, which is the same as taking the square root of each side of the equation.

   ```
   c = (a**2 + b**2) ** (1/2)
   ```

   is the same as

   $c = \sqrt{(a^2 + b^2)}$

2. The float division operator, `/`, caused the result to be a float data type.

All three kids received $20 from Grandma. Come up with variable names for the three kids and assign to each variable the value 20 in one line of code.

```
ted = jane = jeff = 20
```

Mars averages a distance of 142,000,000 miles from the sun. Write this distance in scientific notation as it would be notated in Python.

```
1.42e8
```

or

```
1.42E8
```

There are two vans. Each van can hold eight people. There are 20 people. Using the Python arithmetic operators, write the expression that correctly solves for how many people still need a seat.

```
20 % 8
```

or

```
20 - (2 * 8)
```

Start with `money = 100` and use the augmented assignment operators to write the code for the following. The `100` represents $100.

```
money = 100
money *= 10        #1 Money invest wisely.  There is ten times the money.
money -= 800       #2 An $800 car repair is taken out of money.
money += 300       #3 Increase money by $300 for a paycheck received.
money -= money     #4 All the money fell out of a pocket.  It is all gone!
```

Chapter 4 Practice

Nothing to solve for. Read and code along.

Chapter 5 Practice

1. Because a gamer can earn a free round of play by scoring 4000 or more points, a programmer should include checking if the score is 4000 points. The logical error is in the following line of code.

```
freePlay = points > 4000 or enemyShipsDestroyed > 5
```

The `>` comparison operator should be `>=`. The same logical error is also found when checking for enemy ships destroyed.

2a. `You win: False`

b. `opponents = 0`

c. `False`

d. Yes, when checking to see if a value is `True` with the equality comparison, if the value is `True`, so is the result of the comparison. If the value is `False`, so is the result of the comparison. So, the value is all that is needed.

e. `opponents` is better named `opponentsLeft`. `opponents` could be mistaken for the number of opponents eliminated.

f. integer data type

g. boolean

Chapter 6 Practice

who_wins.py

```
player1 = 100
player2 = 97

print("Player 1 score: " + str(player1))
print("Player 2 score: " + str(player2) + "\n")
if player1 > player2:
  print("Player 1 wins!")
elif player1 < player2:
  print("Player 2 wins!")
else:
  print("Tie game!")
```

Chapter 7 Practice

There is more than one solution for the *String Cheese* exercise.

```
string_cheese = "this is a very silly string  it contains cheese"

new_string = string_cheese[:9].capitalize() + string_cheese[20:27] + ".  "
new_string += string_cheese[29:].capitalize() + "."
string_cheese = new_string

string_cheese
```

The *A Nybble More* section has nothing to solve. Read and code along.

Chapter 8 Practice

🖫 three_guesses_practice.py

```
import random

number = random.randint(1,10)
guess = 0
count = 0

print("If you can guess the number correctly in three tries, you win a prize!")

while 1:
    guess = int(input("Guess a number between 1 and 10: "))

    count += 1

    if guess == number:
        if count == 1:
            print("Great guess!  You win a new car!")
            break
        elif count == 2:
            print("Great guess!  You win $100!")
            break
        elif count == 3:
            print("Great guess!  You win a kazoo!")
            break
    elif count >= 3:
        print("Better luck next time!  The number was " + str(number) + ".")
        break
```

🖫 lucky_seven.py

```
import random

i = 0
total_sevens = 0

while i < 10:   #the for loop is covered later

    roll_1 = random.randint(1,6)
    roll_2 = random.randint(1,6)
    print(str(roll_1) + " " + str(roll_2))

    if roll_1 + roll_2 == 7:
        total_sevens += 1  #total_sevens = total_sevens + 1

    i += 1


print(str(total_sevens) + " sevens were rolled!")
```

Chapter 9 Practice

wish_list.py

```
wish_list = []

print("\nEnter three items you want for Christmas.\n")

wish_list.append(input("Enter item: "))
wish_list.append(input("Enter item: "))
wish_list.append(input("Enter item: "))

print("\nDear Santa, \n\n    The user would like a " + wish_list[0] + ", " +
      wish_list[1] + ", and " + wish_list[2] + " for Christmas. \n\nSincerely,\nthe computer")
```

Random Chore Selector

```
import random
chore_list = ["mow lawn", "fold laundry", "wash dog", "mop floor", "dust"]
chore_selected = random.choice(chore_list)
print("The selected chore is : " + chore_selected)
chore_list.remove(chore_selected)
print("There are " + str(len(chore_list)) + " chores left.")
chore_list
```

Chapter 10 Practice

addition_drill.py

```python
import random

for x in range(10):
    num1 = random.randint(1,10)
    num2 = random.randint(1,10)
    soln = num1 + num2
    answer = -1
    while soln != answer:
        answer = input(str(num1) + " + " + str(num2) + " = ")
        answer = int(answer)
```

Chapter 11 Practice

💾 donut_shop.py

```
donut_price = .75
kolache_price = 1.50

num_donuts = int(input("How many donuts would you like? "))
num_kolaches = int(input("How many kolaches would you like? "))

num_dozens = num_donuts // 12
cost_per_dozen = 12 * .75 * .95
non_dozen = num_donuts % 12
donut_cost = num_dozens * cost_per_dozen + (non_dozen * .75)

kolache_cost = num_kolaches * 1.50

customer_total = (donut_cost + kolache_cost) * 1.08

print(f"\nYour total is ${customer_total:.2f}.")
```

💾 how_many_miles.py

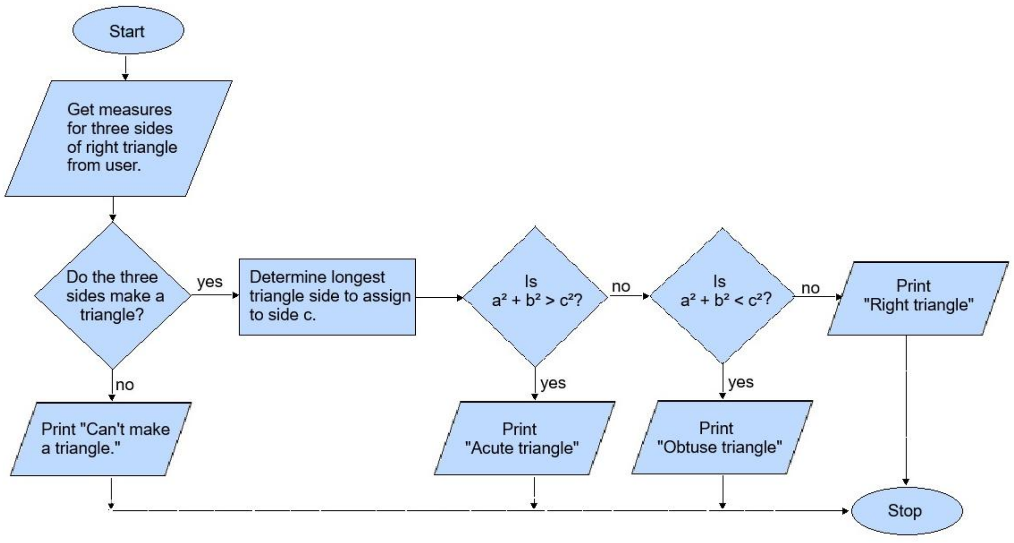
```
light_year = 9460730472581

num_light_years = float(input("How many light years would you like to convert to miles? "))

result = light_year * .6214 * num_light_years

print(f"{num_light_years} light years converts to approximately {result:e} miles.")
```

💾 ad_Lib.py

```
print("Ad-Lib\n")

name = input("Enter a person's name: ")
noun = input("Enter a noun: ")
verb1 = input("Enter a verb: ")
verb2 = input("Enter another verb: ")
verb3 = input("Enter a verb in past tense: ")
adjective = input("Enter an adjective: ")

print(f"\n  The customer walked up to "
      + f"{name} at the counter \nand ordered a "
      + f"{noun}.  The customer sat down at a \ntable and started to "
      + f"{verb1}.  {name} brought out \nthe {noun}"
      + f".  The customer proceeded to {verb2} the {noun}"
      + f".  \nThe customer {verb3} {name} and felt {adjective}.")
```

Chapter 12 Practice

Flowchart


Start
Get measures for three sides of right triangle from user.
Do the three sides make a triangle?
yes
no
Determine longest triangle side to assign to side c.
Is $a^2 + b^2 > c^2$?
no
yes
Is $a^2 + b^2 < c^2$?
no
yes
Print "Right triangle"
Print "Can't make a triangle."
Print "Acute triangle"
Print "Obtuse triangle"
Stop

flowchart for triangle_calc.py

Pseudocode

```
Get measures of three triangle sides from user
If the sides make a triangle
    Make sure the longest side is assigned to variable c
    if $a^2 + b^2 > c^2$
        print the triangle is acute
    if $a^2 + b^2 < c^2$
        print the triangle is obtuse
    otherwise, print the triangle is a right triangle
Else (If the sides did not make a triangle)
    print not a triangle
```

See next page for the rest of the Chapter 14 solutions.

🖫 triangle_calc.py

```python
print("\nLet's find out what kind of triangle you can make!\n")

side1 = float(input("Enter length of the first side in cm: "))
side2 = float(input("Enter length of the second side in cm: "))
side3 = float(input("Enter length of the third side in cm: "))

print("")

if side1 + side2 > side3 and side1 + side3 > side2 and side2 + side3 > side1:
    if side1 >= side2 and side1 >= side3:
        c = side1
        b = side2
        a = side3
    elif side2 >= side1 and side2 >= side3:
        c = side2
        b = side1
        a = side3
    else:
        c = side3
        b = side2
        a = side1
    if a**2 + b**2 > c**2:
        print("You can make an acute triangle.")
    elif a**2 + b**2 < c**2:
        print("You can make an obtuse triangle.")
    else:
        print("You can make a right triangle.")
else:
    print("The side lengths you entered can't make a triangle.")
```

Chapter 13 Practice

💾 honey_do_list.txt

```
change the tire
fix the fence
build a treehouse
```

💾 honey_do.py

```
print("\n\nHoney-Do List\n")

f = open("honey_do_list.txt")
print(f.read())
f.close()

f = open("honey_do_list.txt","a")
for x in range(3):
    task = input("Add a task to the list: ")
    f.write(task + "\n")
f.close()

f = open("honey_do_list.txt")
print("\nUpdated Honey-Do List\n")
print(f.read())
f.close()
```

Chapter 14 Practice

playlist_shuffler.py

```
import random

playlist = ["Best Song Ever", "Love Song", "Sad Song", "Sing Along Song", "Dance Song"]
random.shuffle(playlist)
print(playlist)
```

See next page for the rest of the Chapter 14 solutions.

1. The user chooses the number of words to unscramble. The yellow highlighted code is newly added code. The blue highlighted code is amended.

Word_Jumble_Draft.py

```
import random

f = open("ChristmasWordBank.txt")

actual_words = []
jumbled_words = []

solutions_key = []

i = 0

for x in f:
    actual_words.append(x)
    actual_words[i] = actual_words[i].rstrip().upper()

    jumbled_words.append(actual_words[i])

    tempList = list(jumbled_words[i])
    random.shuffle(tempList)
    jumbled_words[i] = "".join(tempList)

    solutions_key.append([jumbled_words[i],actual_words[i]])

    i  += 1

f.close()

random.shuffle(solutions_key)

print("\n\nUnscramble the Jumbled Christmas Words!\n")

total_words = len(solutions_key)
num_words = int(input("How many words would you like to unscramble? "))

while num_words > total_words:
    print(f"Sorry!  There are only {total_words} total.")
    num_words = int(input("Please enter a new number: "))


for x in range(num_words):    ###amended
    misses = 0
    print(solutions_key[x][0] + "\n")
    answer = input("? ")
    while answer.upper().strip() != solutions_key[x][1]:
        misses += 1
        if misses >= 3:
            print("The word was: " + solutions_key[x][1] + ".")
            break
        answer = input("? ")

    print("\n")

print("The End")
```

2. Allow the user an optional hint on his final attempt at unscrambling the word.

Word_Jumble_Draft.py

```python
import random

f = open("ChristmasWordBank.txt")

actual_words = []
jumbled_words = []

solutions_key = []

i = 0

for x in f:
    actual_words.append(x)
    actual_words[i] = actual_words[i].rstrip().upper()

    jumbled_words.append(actual_words[i])

    tempList = list(jumbled_words[i])
    random.shuffle(tempList)
    jumbled_words[i] = "".join(tempList)

    solutions_key.append([jumbled_words[i],actual_words[i]])

    i  += 1

f.close()

random.shuffle(solutions_key)

print("\n\nUnscramble the Words!\n")

total_words = len(solutions_key)
num_words = int(input("How many words would you like to unscramble? "))

while num_words > total_words:
    print(f"Sorry!  There are only {total_words} total.")
    num_words = int(input("Please enter a new number: "))

for x in range(num_words):
    misses = 0
    print(solutions_key[x][0] + "\n")
    answer = input("? ")
    while answer.upper().strip() != solutions_key[x][1]:
        misses += 1

        if misses == 2:
            wants_help = input("Would you like a hint? ")
            if wants_help == "" or wants_help[0].lower() == "y":
                print(f"The word starts with a {solutions_key[x][1][0]}")

        if misses >= 3:
            print("The word was: " + solutions_key[x][1] + ".")
            break
        answer = input("? ")

    print("\n")

print("The End")
```

3. Create another word bank and edit the starting instruction to the user accordingly.
An EasterWordBank.txt file will need to be created in the same folder for the solution below to work.

Word_Jumble.py

```
import random

f = open("EasterWordBank.txt")    ###amended

actual_words = []
jumbled_words = []

solutions_key = []

i = 0

for x in f:
    actual_words.append(x)
    actual_words[i] = actual_words[i].rstrip().upper()

    jumbled_words.append(actual_words[i])

    tempList = list(jumbled_words[i])
    random.shuffle(tempList)
    jumbled_words[i] = "".join(tempList)

    solutions_key.append([jumbled_words[i],actual_words[i]])

    i  += 1

f.close()

random.shuffle(solutions_key)

print("\n\nUnscramble the Jumbled Words!\n")    ###amended

total_words = len(solutions_key)
num_words = int(input("How many words would you like to unscramble? "))

while num_words > total_words:
    print(f"Sorry!  There are only {total_words} total.")
    num_words = int(input("Please enter a new number: "))

for x in range(num_words):
    misses = 0
    print(solutions_key[x][0] + "\n")
    answer = input("? ")
    while answer.upper().strip() != solutions_key[x][1]:
        misses += 1

        if misses == 2:
            wants_help = input("Would you like a hint? ")
            if wants_help == "" or wants_help[0].lower() == "y":
                print(f"The word starts with a {solutions_key[x][1][0]}")

        if misses >= 3:
            print("The word was: " + solutions_key[x][1] + ".")
            break
        answer = input("? ")

    print("\n")

print("The End")
```

Chapter 15 Practice

country_list.py

```
countries = [
              { "name": "Canada",
                "capital": "Ottowa",
                "major cities": "Montreal and Toronto",
                "continent": "North America",
                "population": "40,100,000"
              },
              { "name": "Germany",
                "capital": "Berlin",
                "major cities": "Frankfurt and Munich",
                "continent": "Europe",
                "population": "84,480,000"
              },
              { "name": "United States of America",
                "capital": "Washington D.C.",
                "major cites": "New York, Los Angeles, and Houston",
                "continent": "North America",
                "population": "334,900,000"
              }
            ]

i = 0
print()
for x in countries:
    for k, v in countries[i].items():
        print(f"{k}: {v}")
    print("\n")
    i += 1
```

Chapter 16 Practice

convert_temp.py

```
def convert_to_fahrenheit(c):
    return c * 9/5 + 32

def convert_to_celsius(f):
    return (f - 32) * 5/9

celsius = float(input("\nEnter the degrees in Celsius to convert to Fahrenheit: "))
print(f"{celsius}° C equals {convert_to_fahrenheit(celsius)}° F.")

fahrenheit = float(input("\nEnter the degrees in Fahrenheit to convert to Celsius: "))
print(f"{fahrenheit}° F equals {convert_to_celsius(fahrenheit)}° C.")
```

stickman_returns.py

```
import turtle
t = turtle.Pen()

def draw_stickman():
    """Using the code from stickman_practice.py from the Chapter 1 Practice section."""
    t.circle(20)
    t.right(90)
    t.forward(65)
    t.left(30)
    t.forward(50)
    t.backward(50)
    t.right(60)
    t.forward(50)
    t.penup()
    t.right(150)
    t.forward(85)
    t.right(90)
    t.pendown()
    t.forward(50)
    t.hideturtle()
```

```
draw_stickman()
```