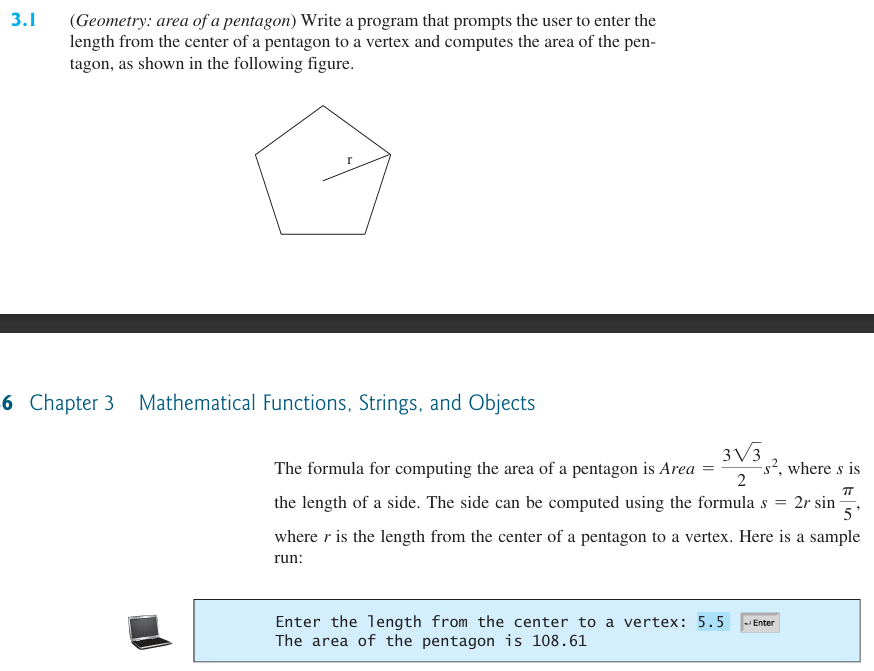
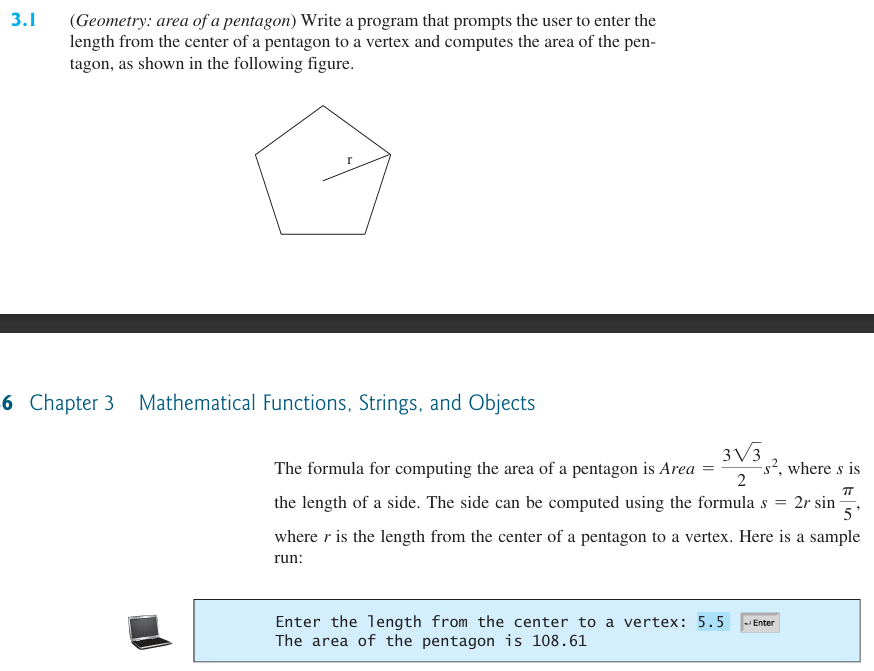
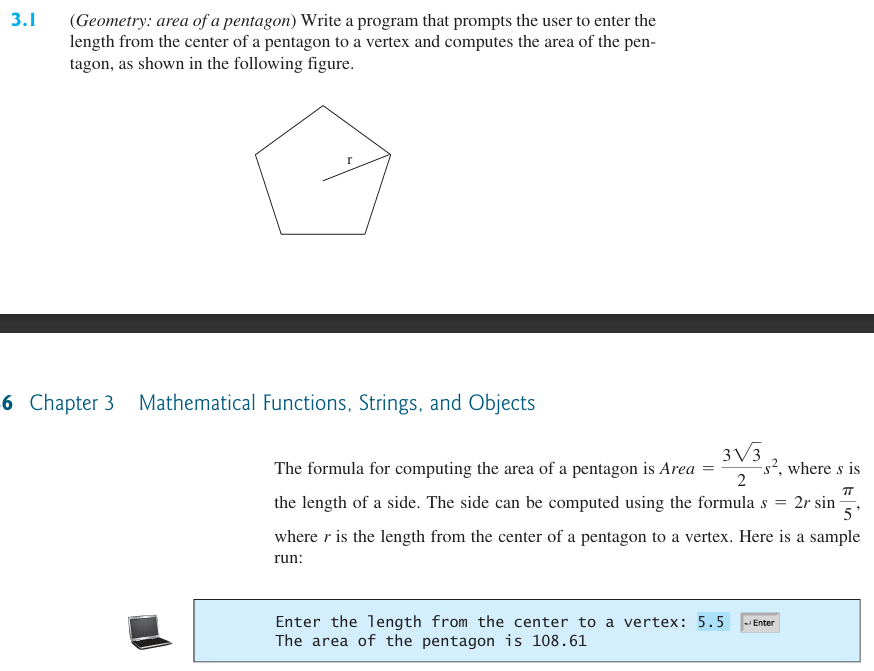
*Date: \_\_/09/2024*

***Assignment 02***

*Name: MEHRIN FARZANA*

*ID: 2101013*

*Problem no. 1.*



*Solution:*

Code:

import math

x=float(input("Enter the length from the center to a vertex:"))

print("The area of the pentagon is", end=" ")

y= round(3\*math.sqrt(3)\*(2\*x\*math.sin(math.pi/5))\*\*2/2, 2)

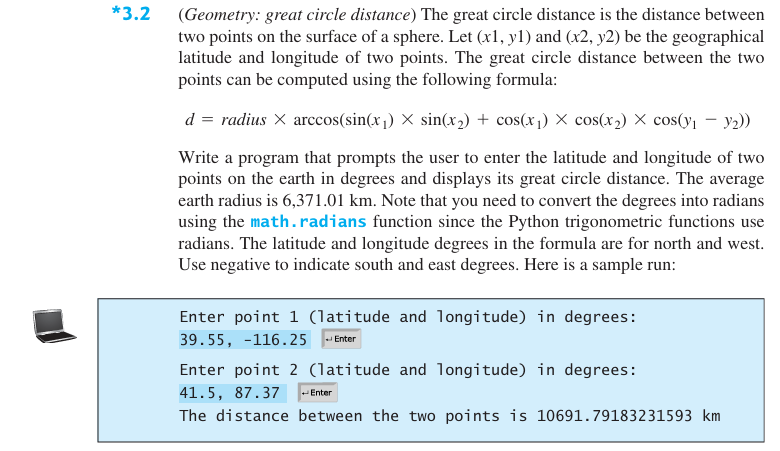
print(y)

Output:



Figure 1.1: Output

*Problem no. 2.*



*Solution:*

Code:

import math

x1, y1 = map(float, input("Enter point 1 (latitude and longitude) in degrees: ").split(","))

x2, y2 = map(float, input("Enter point 2 (latitude and longitude) in degrees: ").split(","))

x1, y1, x2, y2 = math.radians(x1), math.radians(y1), math.radians(x2), math.radians(y2)

d = (6371.01 \* (math.acos(math.sin(x1) \* math.sin(x2) + math.cos(x1) \* math.cos(x2) \* math.cos(y1- y2))))

print("The distance between the two points is ", end=" ")

print(d, end=" ")

print("km")

Output:

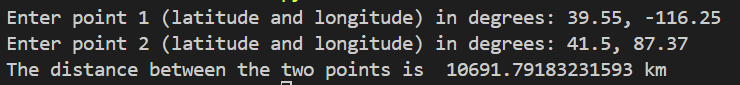
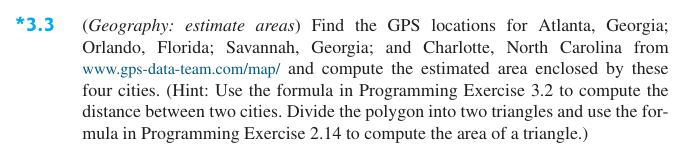


Figure 2.1: Output

*Problem no. 3.*



*Solution:*

Code:

import math

x1, y1= -84.387917 , 33.757222

x2, y2= -81.5197376 , 28.3703828

x3, y3= -81.0983 , 32.0749

x4, y4= -80.8449 , 35.0589

x1, y1, x2, y2, x3, y3, x4, y4= math.radians(x1), math.radians(y1), math.radians(x2), math.radians(y2), math.radians(x3), math.radians(y3), math.radians(x4), math.radians(y4)

d1 = (6371.01 \* (math.acos(math.sin(x1) \* math.sin(x2) + math.cos(x1) \* math.cos(x2) \* math.cos(y1- y2))))

d2 = (6371.01 \* (math.acos(math.sin(x2) \* math.sin(x3) + math.cos(x2) \* math.cos(x3) \* math.cos(y2- y3))))

d3 = (6371.01 \* (math.acos(math.sin(x1) \* math.sin(x3) + math.cos(x1) \* math.cos(x3) \* math.cos(y1- y3))))

d4 = (6371.01 \* (math.acos(math.sin(x1) \* math.sin(x4) + math.cos(x1) \* math.cos(x4) \* math.cos(y1- y4))))

d5 = (6371.01 \* (math.acos(math.sin(x3) \* math.sin(x4) + math.cos(x3) \* math.cos(x4) \* math.cos(y3- y4))))

s1 = (d1 + d2 + d3) / 2

s2 = (d3 + d4 + d5) / 2

a1 = round(math.sqrt(s1\*(s1- d1)\*(s1- d2)\*(s1- d3)), 1)

a2 = round(math.sqrt(s2\*(s2- d3)\*(s2- d4)\*(s2- d5)), 1)

a = a1 + a2

print("Estimated area is", end=" ")

print(a, end=" ")

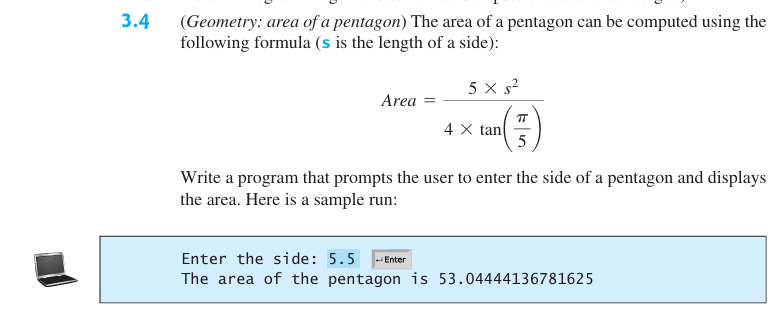
print("km^2")

Output:



Figure 3.1: Output

*Problem no. 4.*



*Solution:*

Code:

import math

s=float(input("Enter the side: "))

area = (5 \* s\*\*2)/ (4 \* math.tan(math.pi/5))

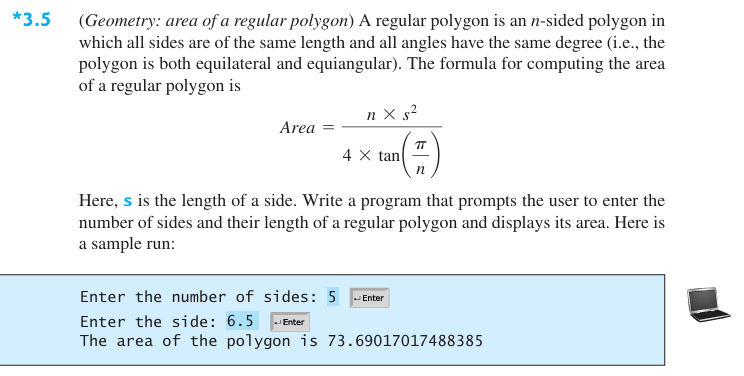
print("The area of the pentagon is", area)

Output:



Figure 4.1: Output

*Problem no. 5.*



*Solution:*

Code:

import math

n=float(input("Enter the number of sides: "))

s=float(input("Enter the side: "))

area = (n \* s\*\*2)/ (4 \* math.tan(math.pi/n))

print("The area of the polygon is", area)

Output:

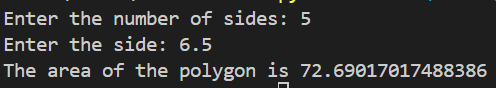
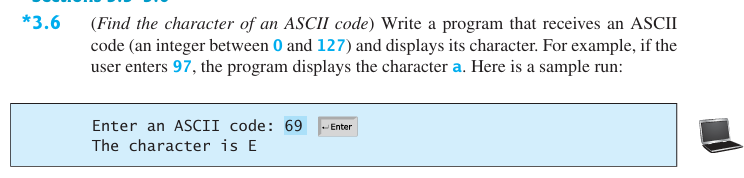


Figure 5.1: Output

*Problem no. 6.*



*Solution:*

Code:

a=int(input("Enter an ASCII code: "))

c=chr(a)

print("The character is ", end="")

print(c)

Output:



Figure 6.1: Output

*Problem no. 7.*



*Solution:*

Code:

import time

import random

time.time()

#a=int(random.random() % 26 + 65)

a=random.randint(65,90)

c=chr(a)

print(c)

Output:

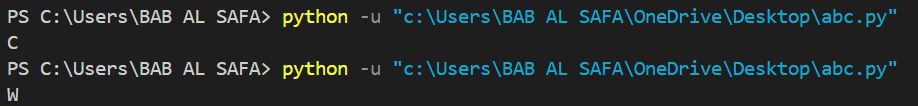
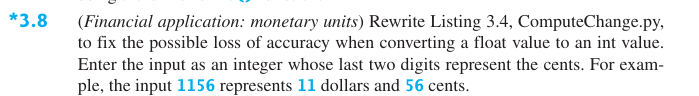


Figure 7.1: Output

*Problem no. 8.*



*Solution:*

Code:

amount = int(input("Enter an amount, for example, 1156 represents 11 dollars and 56 cents: "))

numberOfOneDollars = amount // 100

remainingAmount = amount % 100

numberOfQuarters = remainingAmount // 25

remainingAmount = remainingAmount % 25

numberOfDimes = remainingAmount // 10

remainingAmount = remainingAmount % 10

numberOfNickels = remainingAmount // 5

remainingAmount = remainingAmount % 5

numberOfPennies = remainingAmount

print("Your amount", amount, "consists of\n",

"\t", numberOfOneDollars, "dollars\n",

"\t", numberOfQuarters, "quarters\n",

"\t", numberOfDimes,  "dimes\n",

"\t", numberOfNickels, "nickels\n",

"\t", numberOfPennies, "pennies")

Output:

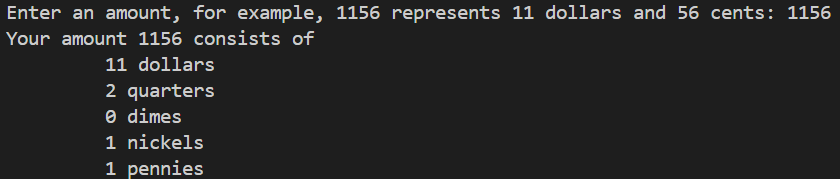
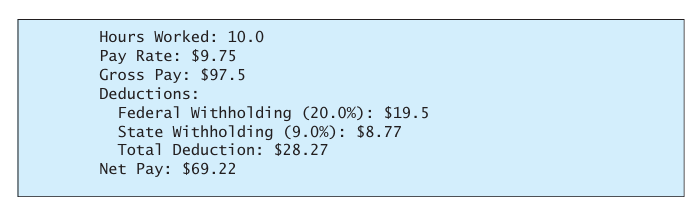
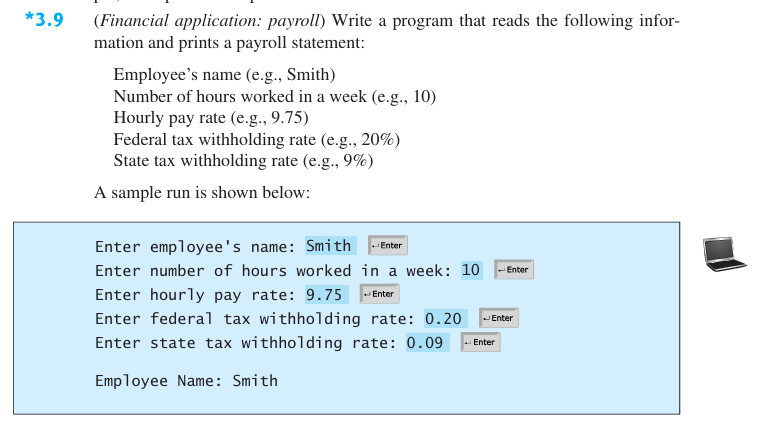


Figure 8.1: Output

*Problem no. 9.*



*Solution:*

Code:

name = input("Enter employee's name: ")

work = float(input("Enter number of hours worked in a week: "))

rate = float(input("Enter hourly pay rate: "))

fedTax = float(input("Enter federal tax withholding rate: "))

stTax = float(input("Enter state tax withholding rate: "))

print("Employee Name: ", end="")

print(name)

print("Hours Worked: ", end ="")

print(work)

print("Pay Rate: $", end ="")

print(rate)

print("Gross Pay: $", end ="")

grossPay=round(work\*rate, 2)

print(grossPay)

print("Deductions:")

print("Federal Withholding (", end="")

print(fedTax\*100, end="")

print("%): $", end="")

fedTaxDed = round(grossPay\*fedTax, 2)

print(fedTaxDed)

print("State Withholding (", end="")

print(stTax\*100, end="")

print("%): $", end="")

stTaxDed = round(grossPay\*stTax, 2)

print(stTaxDed)

print("Total Deduction: $", end="")

print(fedTaxDed + stTaxDed)

print("Net Pay: $", end="")

print(grossPay - (fedTaxDed + stTaxDed))

Output:

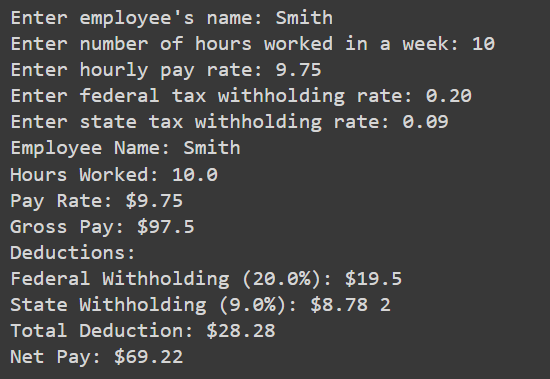
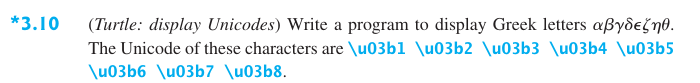


Figure 9.1: Output

*Problem no. 10.*



*Solution:*

Code:

import turtle

screen = turtle.Screen()

screen.title("Greek Letters Display")

pen = turtle.Turtle()

pen.hideturtle()

pen.penup()

greek\_letters = ['\u03b1', '\u03b2', '\u03b3', '\u03b4', '\u03b5', '\u03b6', '\u03b7', '\u03b8']

positions = [(-200, 0), (-150, 0), (-100, 0), (-50, 0), (0, 0), (50, 0), (100, 0), (150, 0)]

for letter, position in zip(greek\_letters, positions):

    pen.goto(position)

    pen.write(letter, font=("Arial", 40, "normal"))

turtle.done()

Output:

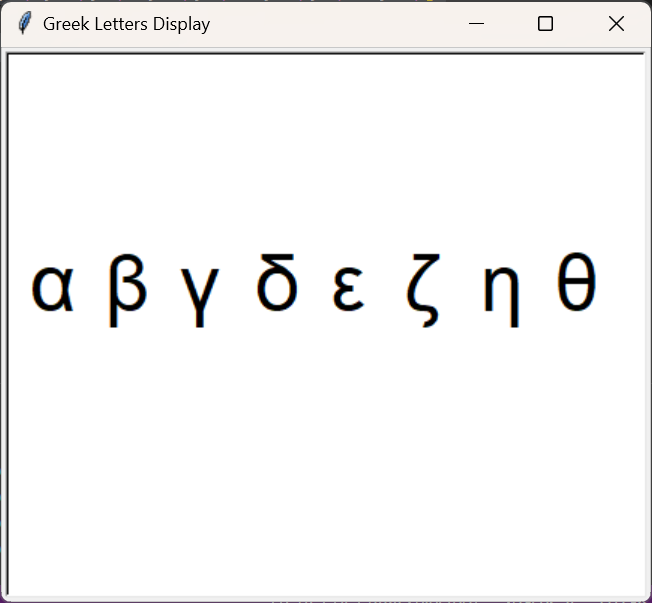
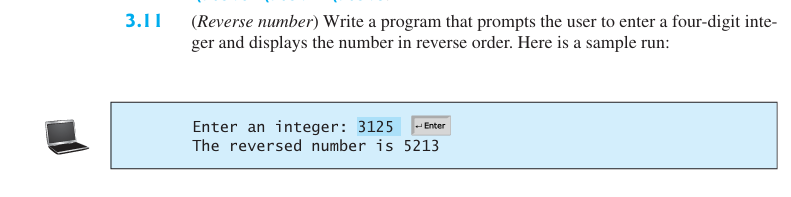


Figure 10.1: Output

*Problem no. 11.*



*Solution:*

Code:

n = int(input("Enter an integer: "))

print("The reversed number is", end=" ")

while(n>0):

 print(n%10, end="")

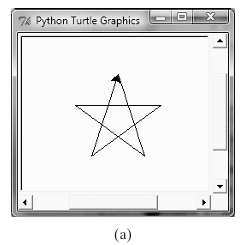
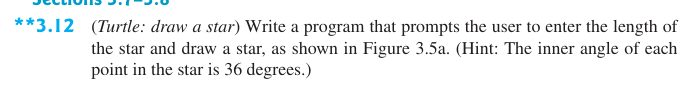
 n //=10

Output:



Figure 11.1: Output

*Problem no. 12.*



*Solution:*

Code:

import turtle

def draw\_star(length):

    angle = 144

    for \_ in range(5):

        turtle.right(angle)

        turtle.forward(length)

turtle.speed(1)

length = int(input("Enter the length of the star: "))

draw\_star(length)

turtle.done()

Output:

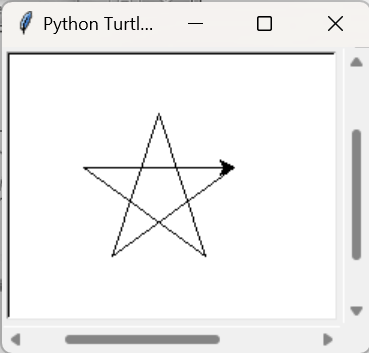


Figure 12.1: Output