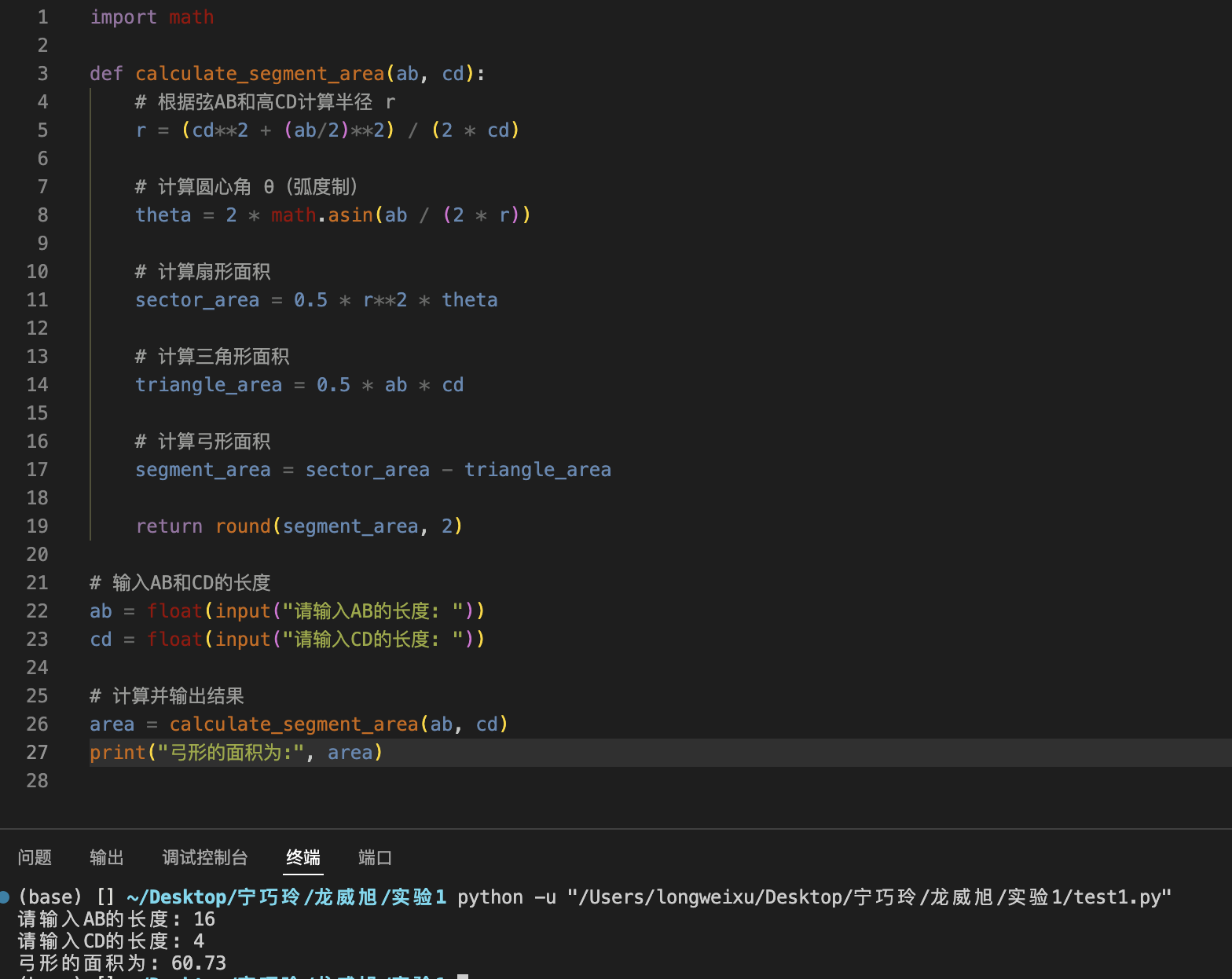
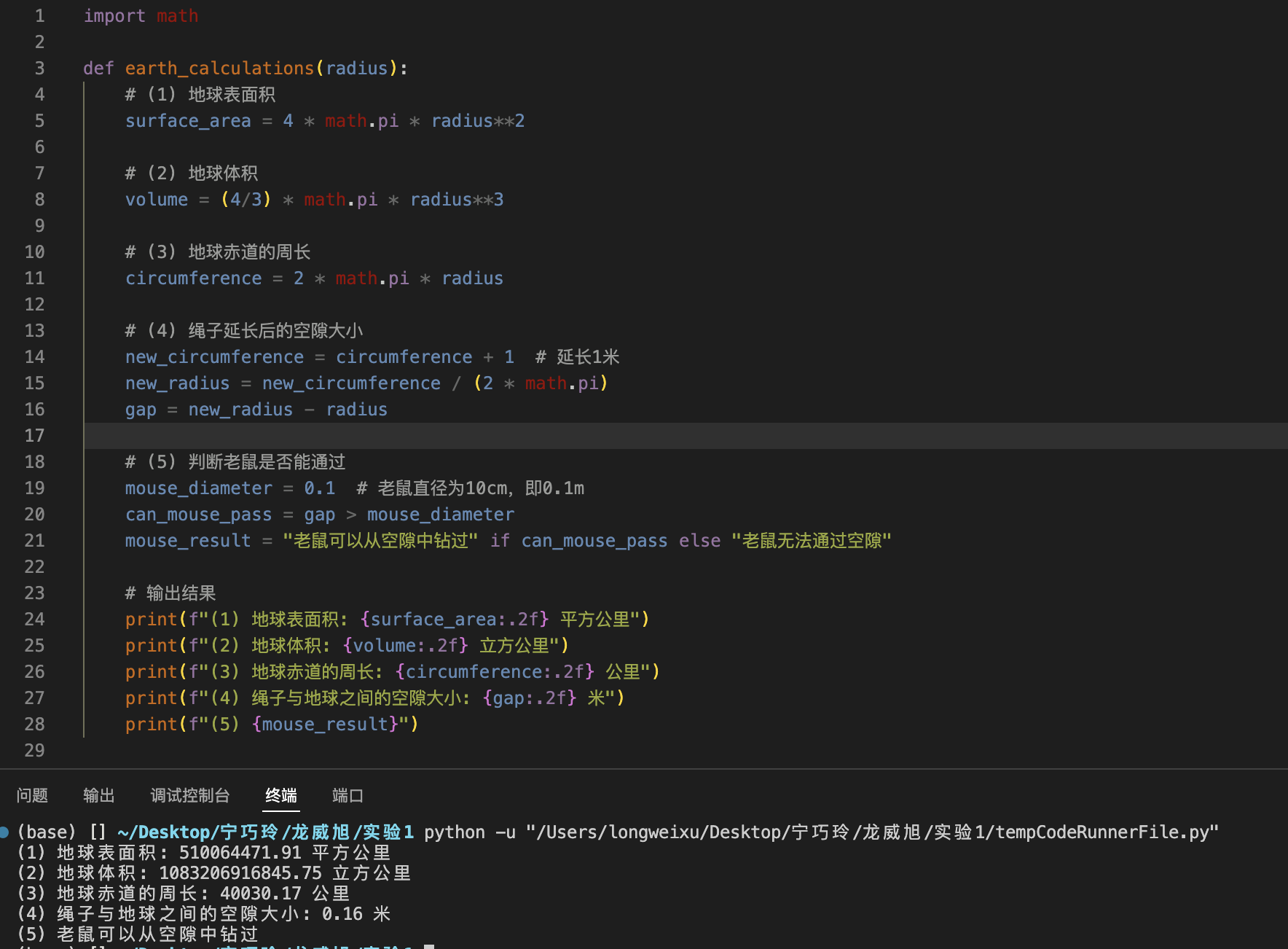
1.AB 是圆的一条弦，ABC形成一个弓形，从键盘分别输人AB和CD 的长度，计算弓形的面积，结果保留小数点后2位有效数字。

1. **import** math
3. **def** calculate\_segment\_area(ab, cd):
4. # 根据弦AB和高CD计算半径 r
5. r = (cd\*\*2 + (ab/2)\*\*2) / (2 \* cd)
6. # 计算圆心角 θ（弧度制）
7. theta = 2 \* math.asin(ab / (2 \* r))
8. # 计算扇形面积
9. sector\_area = 0.5 \* r\*\*2 \* theta
10. # 计算三角形面积
11. triangle\_area = 0.5 \* ab \* cd
13. # 计算弓形面积
14. segment\_area = sector\_area - triangle\_area
16. **return** round(segment\_area, 2)
18. # 输入AB和CD的长度
19. ab = float(input("请输入AB的长度: "))
20. cd = float(input("请输入CD的长度: "))
21. # 计算并输出结果
22. area = calculate\_segment\_area(ab, cd)
23. **print**("弓形的面积为:", area)



2.地球的半径是6371km，假设地球是一个规则的球体，完成以下计算并输出结果，结果保留2位小数。pi 值引用 math 库中常数 pi 的值。（1）计算地球表面积；（2） 计算地球体积；（3）计算地球赤道的周长；（4）假设有一根绳子正好可以紧贴地球绕赤道一周，紧密地捆绑住地球。现在将绳子延长1m，仍围绕地球赤道一周，假设绳子与地球表面间空隙均匀，计算绳子与地球之间的空隙大小；（5） 判断一只正常大小的老鼠能否从这个空隙穿过？可假设老鼠身体截面为圆柱，且圆柱最粗处直径 10cm，若空隙大于 10cm，则输出“老鼠可以从空隙中钻过”，否则输出“老鼠无法通过空隙”。

1. **import** math
3. **def** earth\_calculations(radius):
4. # (1) 地球表面积
5. surface\_area = 4 \* math.pi \* radius\*\*2
7. # (2) 地球体积
8. volume = (4/3) \* math.pi \* radius\*\*3
10. # (3) 地球赤道的周长
11. circumference = 2 \* math.pi \* radius
13. # (4) 绳子延长后的空隙大小
14. new\_circumference = circumference + 1  # 延长1米
15. new\_radius = new\_circumference / (2 \* math.pi)
16. gap = new\_radius - radius
18. # (5) 判断老鼠是否能通过
19. mouse\_diameter = 0.1  # 老鼠直径为10cm，即0.1m
20. can\_mouse\_pass = gap > mouse\_diameter
21. mouse\_result = "老鼠可以从空隙中钻过" **if** can\_mouse\_pass **else** "老鼠无法通过空隙"
23. # 输出结果
24. **print**(f"(1) 地球表面积: {surface\_area:.2f} 平方公里")
25. **print**(f"(2) 地球体积: {volume:.2f} 立方公里")
26. **print**(f"(3) 地球赤道的周长: {circumference:.2f} 公里")
27. **print**(f"(4) 绳子与地球之间的空隙大小: {gap:.2f} 米")
28. **print**(f"(5) {mouse\_result}")
30. # 地球半径
31. earth\_radius = 6371  # 单位为公里
32. earth\_calculations(earth\_radius)



3. 读文本文件 my。txt完成如下任务：（1）统计文档中大写字母、小写字母、 数字、空格和其他字符的数量；（2）统计文中单词的数量，形如Its等勇’缩写的两个单词按2个单词计数；（3） 取出每个单词的首字母拼接成一个字符串。

1. **import** string
3. **def** analyze\_text(filename):
4. uppercase\_count = 0
5. lowercase\_count = 0
6. digit\_count = 0
7. space\_count = 0
8. other\_count = 0
9. word\_count = 0
10. first\_letter\_string = ""
12. with open(filename, 'r', encoding='utf-8') as file:
13. text = file.read()
15. # (1) 统计大写字母、小写字母、数字、空格和其他字符的数量
16. **for** char **in** text:
17. **if** char.isupper():
18. uppercase\_count += 1
19. **elif** char.islower():
20. lowercase\_count += 1
21. **elif** char.isdigit():
22. digit\_count += 1
23. **elif** char.isspace():
24. space\_count += 1
25. **else**:
26. other\_count += 1
28. # (2) 统计单词数量，缩写的两个单词按2个单词计数
29. words = text.replace("’", "'""'").replace("'", " ").split()
30. word\_count = len(words)
32. # (3) 拼接每个单词的首字母
33. **for** word **in** words:
34. **if** word:  # 确保不为空
35. first\_letter\_string += word[0]
37. # 输出结果
38. **print**(f"(1) 大写字母数量: {uppercase\_count}")
39. **print**(f"    小写字母数量: {lowercase\_count}")
40. **print**(f"    数字数量: {digit\_count}")
41. **print**(f"    空格数量: {space\_count}")
42. **print**(f"    其他字符数量: {other\_count}")
43. **print**(f"(2) 单词数量: {word\_count}")
44. **print**(f"(3) 首字母拼接字符串: {first\_letter\_string}")
46. # 调用函数，分析文件 my.txt
47. analyze\_text("my.txt")

