Task 2: software design and development

In the board game ‘Capturing Olympus’, six players work as a team to earn points. One point is earned if the six players score a combined total of more than 50 hits. An additional point is earned if the average number of hits is greater than or equal to 10.

Read the following analysis and design carefully.

Program analysis

A program is required to determine the number of points earned by the team. The program will ask the user to enter the number of hits scored by each of the six players and store these values. When all six players’ hits have been entered, the program will calculate the total and average number of hits. A message indicating the points earned is then displayed to the user.

Inputs

- a valid number of hits scored by each of the six players

Processes

- calculate the total hits achieved by all six players
- calculate an average number of hits (total/6)
- determine if the six players have earned points

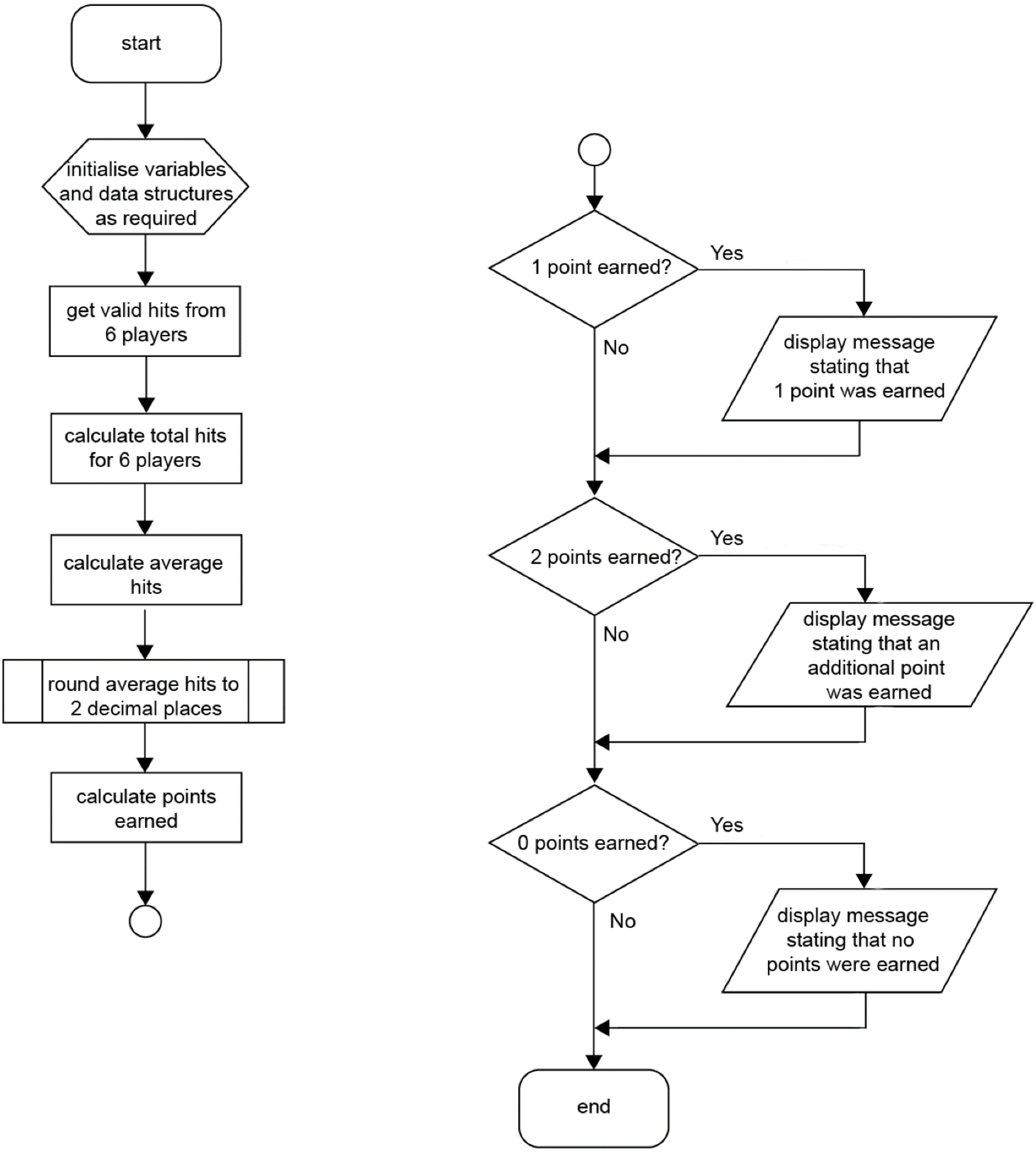
Outputs

- a message is displayed if one point has been earned
- a message is displayed if the additional point has been earned
- a message is displayed if no points have been earned

Assumptions

- the number of hits a single player can achieve is greater than or equal to 0 and less than or equal to 30
- the average should be displayed to two decimal places
- one point is earned if the total number of hits is greater than 50. An additional point is earned if the average number of hits is greater than or equal to 10

Program design (flow chart)

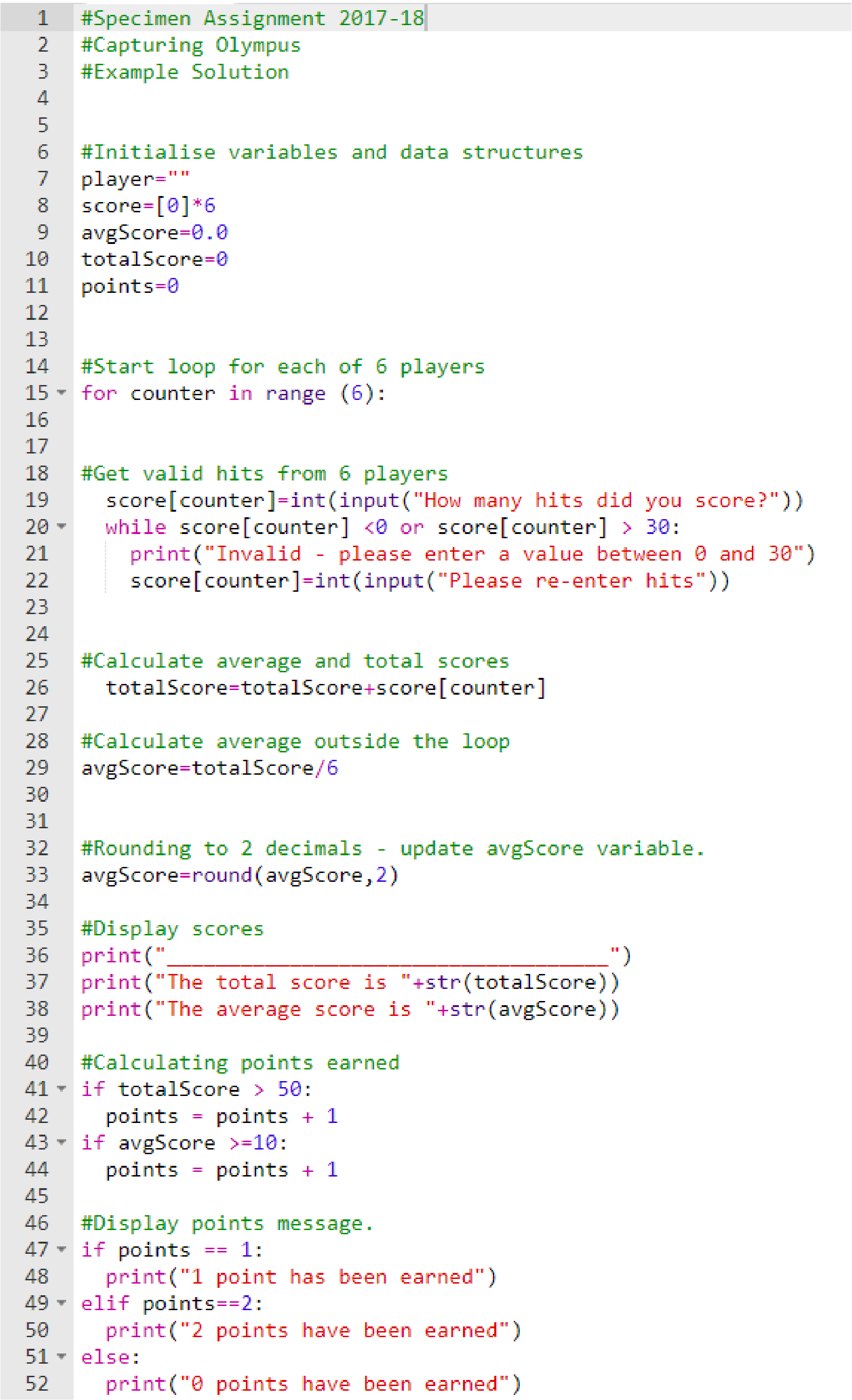

start
initialise variables and data structures as required
get valid hits from 6 players
calculate total hits for 6 players
calculate average hits
round average hits to 2 decimal places
calculate points earned
1 point earned?
Yes
No
display message stating that 1 point was earned
2 points earned?
Yes
No
display message stating that an additional point was earned
0 points earned?
Yes
No
display message stating that no points were earned
end

Task 2: software design and development

2a Using the program analysis and flowchart design, implement the program in a language of your choice. Ensure the program matches the design.

(15 marks)

Print evidence of the program code.



```
#Specimen Assignment 2017-18
#Capturing Olympus
#Example Solution


#Initialise variables and data structures
player=""
score=[0]*6
avgScore=0.0
totalScore=0
points=0


#Start loop for each of 6 players
for counter in range (6):


#Get valid hits from 6 players
  score[counter]=int(input("How many hits did you score?"))
  while score[counter] <0 or score[counter] > 30:
    print("Invalid - please enter a value between 0 and 30")
    score[counter]=int(input("Please re-enter hits"))


#Calculate average and total scores
  totalScore=totalScore+score[counter]

#Calculate average outside the loop
avgScore=totalScore/6


#Rounding to 2 decimals - update avgScore variable.
avgScore=round(avgScore,2)

#Display scores
print("____________________________________")
print("The total score is "+str(totalScore))
print("The average score is "+str(avgScore))

#Calculating points earned
if totalScore > 50:
  points = points + 1
if avgScore >=10:
  points = points + 1

#Display points message.
if points == 1:
  print("1 point has been earned")
elif points==2:
  print("2 points have been earned")
else:
  print("0 points have been earned")
```

2b Complete the table below to create two sets of test data. You must demonstrate that the program correctly outputs the messages that one or both points have been earned.

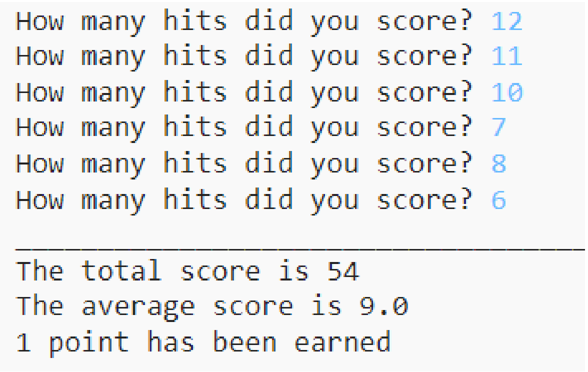
(3 marks)

Type of test	Input	Expected output	Actual output
Normal	Player 1 12 Player 2 11 Player 3 10 Player 4 7 Player 5 8 Player 6 6	Program displays message stating one point earned.	Attach printouts of inputs and outputs as evidence.
Normal	Player 1 12 Player 2 12 Player 3 17 Player 4 18 Player 5 19 Player 6 23	Program displays message stating two points earned.	Attach printouts of inputs and outputs as evidence.

Test your program using both sets of test data. Print evidence of inputs and outputs to show that you have completed each test.

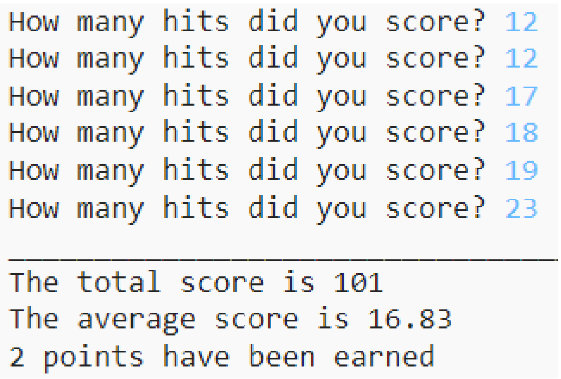
Evidence:

Test 1



```
How many hits did you score? 12
How many hits did you score? 11
How many hits did you score? 10
How many hits did you score? 7
How many hits did you score? 8
How many hits did you score? 6
___________________________________
The total score is 54
The average score is 9.0
1 point has been earned
```

Test 2



```
How many hits did you score? 12
How many hits did you score? 12
How many hits did you score? 17
How many hits did you score? 18
How many hits did you score? 19
How many hits did you score? 23
___________________________________
The total score is 101
The average score is 16.83
2 points have been earned
```

2c The program should ensure that only a valid number of hits can be entered for each of the six players.

State **two** extreme and **one** exceptional numerical value that could be used as part of a test run to check that only a valid number of hits can be entered:

(2 marks)

Extreme 1: 0 Extreme 2: 30

Exceptional : -10

Candidate name______________________ Candidate number___________

2d Evaluate your program by commenting on the following:

Fitness for purpose **(1 mark)**

My program is fit for purpose as it carries out all the requirements of the task.
My program successfully:

- Allows the user to input hits for 6 players. This was achieved by using a fixed loop.
- Stores the values for these in an array
- Validates the input between 0 and 30 using a conditional loop
- Keeps a running total of the combined hits for all 6 players
- Calculates the average
- Accurately displays how many points they have received - 1 point for total hits over 50 and an additional point for average over or equal to 10.
- I have thoroughly tested my program using Normal, Extreme and Exceptional results and it produces accurate results.

Efficiency of your code **(1 mark)**

My code is efficient as I have used an array to store the values instead of 6 single variables. This allowed me to keep a running total of hits for all six players using a fixed loop

I also used an IF…ELIF….ELSE to reduce the number of IF statements to 2 instead of 3 which means the processor has less comparisons to make.

Robustness of your completed program **(1 mark)**

I have tested my program using Normal, Extreme and Exceptional data and it coped well with unexpected inputs.

Extreme test data 0 and 30 was accepted as valid.

Exceptional test data.

- -10 was input and the user was prompted to re-enter as it was invalid.
- 45 was entered and they were asked to re- enter.
- I tried to input ‘A’ and my program crashed, this was expected though as validating a character is out with the scope for National 5.

Readability of your code **(2 marks)**
My code is readable as I have used good programming techniques.
I have used:

- Meaningful variable names
- Good use of white space. Python uses indentation by default but this also helps readability.
- Internal commentary throughout to explain the main parts of my code. This could help with maintenance in the future.

Candidate name__________________________ Candidate number_______________