Teaching guide: Programming challenge 2 Student scores

Programming is a fundamental skill required for success in GCSE Computer Science. This programming challenge is designed to develop students' programming skills.

In the paper 1 exam, students will be required to design, write, test and refine program code in either C#, Python (version 3) or VB.Net.

To develop their programming skills, students should have sufficient practical experience of:

- structuring programs into modular parts with clear documented interfaces to enable them to design appropriate modular structures for solutions
- including authentication and data validation systems/routines within their computer programs
- writing, debugging and testing programs to enable them to develop the skills to articulate how programs work and argue using logical reasoning for the correctness of programs in solving specified problems
- designing and applying test data (normal, boundary and erroneous) to the testing of programs so that they are familiar with these test data types and the purpose of testing
- refining programs in response to testing outcomes.

Programming Challenge 2 - Student scores

Exercise 1

This exercise can be solved using iteration. The validation could be implemented in different ways. The basic example solution in **Figure 1** uses the `int` function, so the program will crash if anything other than an integer is entered. Simple `IF` statements are then used to check the range of the number and if an invalid input is found the program stops. Students can be encouraged to think how to create a program that will produce an error message but not crash or stop. They can also explain how they would test the program.

Another example solution is shown in **Figure 2**, which uses a Python function to validate the input and check for a number, and a nested `WHILE` loop to keep the program running until correct input is entered.

Figure 3 shows an example using a subroutine to validate input.

Specification coverage: sections 3.2.1, 3.2.6, 3.2.7, 3.2.10 and 3.2.11

Figure 1

```
##########################
#   AQA Student Scores    #
##########################

print("Welcome to AQA Student Scores")

runningTotal = 0

print("Please enter 10 student scores")

# Loop that will be executed 10 times only
for i in range(10):
    userInput = int(input("Student score: "))
    if userInput > 10:
        print("Your number must be 10 or less")
        break
    if userInput < 0:
        print("Your number must be 0 or greater")
        break

    runningTotal = runningTotal + userInput
# End of for loop

average = runningTotal / 10
print("Average = " + str(average))

# End of program
```

Figure 2

```
##########################
#  AQA Student Scores    #
##########################

print("Welcome to AQA Student Scores")

runningTotal = 0

print("Please enter 10 student scores")

# Loop that will be executed 10 times only
for i in range(10):
    while True:
        userInput = input("Student score: ")
        if not userInput.isnumeric():
            print("You must enter a positive whole number")
        else:
            score = int(userInput)
            if (score >= 0) and (score <= 10):
                runningTotal = runningTotal + score
                break
            else:
                print("Your number must be between 0 and 10")
    # End of while loop
# End of for loop

average = runningTotal / 10
print("Average = " + str(average))

# End of program
```

Figure 3

```
##########################
#  AQA Student Scores    #
##########################

def validate(userInput):
    while True:
        if not userInput.isnumeric():
            print("You must enter a positive whole number")
        else:
            score = int(userInput)
            if (score >= 0) and (score <= 10):
                return score
            else:
                print("Your number must be between 0 and 10")
        userInput = input("Student score: ")
    # End of while loop

print("Welcome to AQA Student Scores")

runningTotal = 0

print("Please enter up to 10 student scores ")

# Loop that will be executed 10 times only
for i in range(10):
    userInput = input("Student score: ")
    runningTotal = runningTotal + validate(userInput)
# End of for loop

average = runningTotal / 10
print("Average = " + str(average))

# End of program
```

Exercise 2

The example solution in **Figure 4** was used to produce the output in Exercise 2. Students may modify any version of the program created in Exercise 1, or may produce a completely new program, with a menu.

Specification coverage: section 3.2.11

Figure 4

```
##########################
#  AQA Student Scores    #
##########################

print("Welcome to AQA Student Scores")

runningTotal = 0
scoreCount = 0

print("Please enter your student scores or x for the average")

# Loop that will be executed until x is entered
while True:
    userInput = input("Student score: ")
    if userInput == "x":
        break
    else:
        if not userInput.isnumeric():
            print("You must enter a positive whole number")
        else:
            score = int(userInput)
            if (score >= 0) and (score <= 10):
                runningTotal = runningTotal + score
                scoreCount = scoreCount + 1
            else:
                print ("Your number must be between 0 and 10")
# End of while loop

if scoreCount > 0:
    average = runningTotal / scoreCount
    print("Average = " + str(average))
else:
    print("No scores entered")

# End of program
```

Exercise 3

The student can take any version of the program they have developed. This gives an opportunity to look at conditional statements and logic. **Figure 5** is an example using a combination of code shown in **Figure 3** and **Figure 4**. This program is a good example of how developing a trace table can help in designing and testing a program.

Specification coverage: sections 3.2.2 and 3.2.11

Figure 5

```
##########################
#  AQA Student Scores    #
##########################

def validate():
    while True:
        userInput = input("Student score: ")
        if userInput == "x":
            return STOP_CODE
        else:
            if not userInput.isnumeric():
                print("You must enter a positive whole number")
            else:
                score = int(userInput)
                if (score >= 0) and (score <= 10):
                    return score
                else:
                    print("Your number must be between 0 and 10")

    # End of while loop

print("Welcome to AQA Student Scores")

runningTotal = 0
scoreCount = 0
lowestScore = 11
highestScore = -1

# Create a ‘constant’ for indicating when data entry has ended.
# Note in Python true constants do not exist so it is conventional
# to create a variable with an identifier in all caps to signify
# it is a constant value that does not change and should not be
# changed in the program

STOP_CODE = -2

print("Please enter your student scores or x to calculate average")
```

```python
# Loop that will be executed until x is entered
while True:
    score = validate()
    if score == STOP_CODE:
        break
    else:
        userInput = input("Please enter the student first name: ")
        if score > highestScore:
            highestName = userInput
            highestScore = score
        if score < lowestScore:
            lowestName = userInput
            lowestScore = score
        runningTotal = runningTotal + score
        scoreCount = scoreCount + 1

# End of while loop

if scoreCount == 0:
    print("No student scores have been entered")
else:
    average = runningTotal / scoreCount
    print("The student with the highest score is " + highestName)
    print("with " + str(highestScore))
    print("The student with the lowest score is " + lowestName)
    print("with " + str(lowestScore))
    print("Average = " + str(average))

# End of program
```

Extension

The C# and VB.NET examples below use structures to store the student data as records. In Python there are many options for creating a record: lists, dictionaries, tuples and classes. Our preferred method would be to create a new class called `Student`, as covered in the teaching guide: Data Structures (Records). Even though object orientation is not on the GCSE specification this method is very simple to understand and teach and pupils do not need to know anything about OOP to use it in this context.

The example in **Figure 6** shows an example of how this could be implemented in Python. **Figures 7** and **8** show how the same program might be implemented in VB.NET and C#

Figure 6 (Python 3 Version)

```
##########################
#  AQA Student Scores    #
##########################

class Student():
    def __init__(self, name, score):
      self.name = name
      self.score = score

def validate():
    while True:
        userInput = input("Student score: ")
        if userInput == "x":
            return STOP_CODE
        else:
            if not userInput.isnumeric():
                print("You must enter a whole number")
            else:
                score = int(userInput)
                if (score >= 0) and (score <= 10):
                    return score
                else:
                    print("Your number must be between 0 and 10")

def printStudents(students):
    for student in students:
        print('\n' + student.name + " has score " + str(student.score))
```

```
def getScore(student):
    return int(student.score)

def findScore(studentName, students):
    for student in students:
        if student.name == studentName:
            return student.score

    return STOP_CODE

def addScores():
    students = []
    print("Please enter your student scores or x to end")

    while True:
        score = validate()
        if score == STOP_CODE:
            break
        else:
            name = input("Enter the student first name: ")
            students.append(Student(name, score))
    return students

# Main program

STOP_CODE = -2

print("Welcome to AQA Student Scores")

while True:
    print("")
    print("#################################################")
    print("Please enter a menu choice")
    print("Enter 1 to enter student scores")
    print("Enter 2 to sort and print student scores")
    print("Enter 3 to find a student's score")
    print("Enter 4 to quit the program")
    print("#################################################")
    menuItem = input("Menu choice: ")
    if menuItem == '1':
        print("Add scores")
        students = addScores()
        print("Added Students: ")
        printStudents(students)
    elif menuItem == '2':
        students.sort(key=getScore)
        print("Sorted students")
        printStudents(students)
    elif menuItem == '3':
        studentName = input("Enter student name to find: ")
        studentScore = findScore(studentName, students)
        if studentScore != STOP_CODE:
            print("Student " + studentName)
            print("has a score of " + str(studentScore))
        else:
            print(studentName + " not found")
    elif menuItem == '4':
        break
    else:
        print("Unknown option selected!")
```

Figure 7 (VB.NET Version)

```
Imports System

Module Program
  '#########################
  '#  AQA Student Scores    #
  '#########################

  Const STOP_CODE = -2
  Const MAX_STUDENTS = 100

  Structure Student
    Dim name As String
    Dim score As Integer
  End Structure

  Dim students(MAX_STUDENTS) As Student

  Function validate() As Integer
    Do
      Console.Write("Student score: ")
      Dim userInput As String = Console.ReadLine()
      If userInput = "x" Then
        Return STOP_CODE
      Else
        If Not userInput.All(AddressOf Char.IsDigit) Or userInput = "" Then
          Console.WriteLine("You must enter a positive whole number")
        Else
          Dim score As Integer = Convert.ToInt32(userInput)
          If (score >= 0) And (score <= 10) Then
            Return score
          Else
            Console.WriteLine("Your number must be between 0 and 10")
          End If
        End If
      End If
    Loop
  End Function

  Sub printStudents(students() As Student)
    For Each student In students
      If student.name <> "" Then
        Console.WriteLine(Environment.NewLine + student.name + " has score " + student.score.ToString())
      End If
    Next
  End Sub

  Sub addScores(students() As Student)
    Dim scoreCount As Integer = 0

    Console.WriteLine("Please enter your student scores or x to end")

    Do
      Dim score As Integer = validate()
      If score = STOP_CODE Then
        Exit Do
      Else
        Console.Write("Enter the student first name: ")
        Dim studentInput As String = Console.ReadLine()
        students(scoreCount).name = studentInput
        students(scoreCount).score = score
      End If
      scoreCount = scoreCount + 1
    Loop
  End Sub
```

```
Function getScore(stud As Student) As Integer
  Return Convert.ToInt32(stud.score)
End Function

Function findScore(studentName As String, students() As Student) As Integer
  For Each student In students
    If student.name = studentName Then
      Return student.score
    End If
  Next
  Return STOP_CODE
End Function

'Main program

Sub Main(args() As String)
  Console.WriteLine("Welcome to AQA Student Scores")

  Do
    Console.WriteLine("")

    Console.WriteLine("########################################")
    Console.WriteLine("Please enter a menu choice")
    Console.WriteLine("Enter 1 to enter student scores")
    Console.WriteLine("Enter 2 to sort and print student scores")
    Console.WriteLine("Enter 3 to find a student's score")
    Console.WriteLine("Enter 4 to quit the program")

    Console.WriteLine("########################################")
    Console.Write("Menu choice: ")
    Dim menuItem As String = Console.ReadLine()
    Select Case menuItem
```

```
        Case "1"
          Console.WriteLine("Add scores")
          addScores(students)
          Console.WriteLine("Added students: ")
          printStudents(students)
        Case "2"
          students = students.OrderBy(Function(c) c.score).ToArray()
          Console.WriteLine("Sorted students")
          printStudents(students)
        Case "3"
          Console.Write("Enter student name to find: ")
          Dim studentName As String = Console.ReadLine()
          Dim studentScore As Integer = findScore(studentName, students)
          If studentScore <> STOP_CODE Then
            Console.Write("Student " + studentName)
            Console.WriteLine(" has a score of" + Str(studentScore))
          Else
            Console.WriteLine(studentName + " not found")
          End If
        Case "4"
          Exit Do
        Case Else
          Console.WriteLine("Unknown option selected!")
      End Select
    Loop
  End Sub
End Module
```

Figure 8 (C# Version)

```
using System;
using System.Linq; // This module is required in C# but not in VB.NET

namespace CS_StudentScores
{
    class Program
    {
        // #########################
        // #  AQA Student Scores   #
        // #########################

        public struct Student
        {
            public string name;
            public int score;

            public Student(string name, int score)
            {
                this.name = name;
                this.score = score;
            }
        }

        const int STOP_CODE = -2;
        const int MAX_STUDENTS = 100;

        static Student[] students = new Student[MAX_STUDENTS];

        static public int Validate()
        {
            while (true)
            {
                Console.Write("Student score: ");
                string userInput = Console.ReadLine();
                if (userInput == "x")
                {
                    return STOP_CODE;
                }
                else
                {
                    if ((!userInput.All(char.IsDigit)) || (userInput == ""))
                    {
                        Console.WriteLine("You must enter a positive whole number");
                    }
                    else
                    {
                        int score = Convert.ToInt32(userInput);
                        if ((score >= 0) && (score <= 10))
                        {
                            return score;
                        }
                        else
                        {
                            Console.WriteLine("Your number must be between 0 and 10");
                        }
                    }
                }
            }
        }

        static public void PrintStudents(Student[] students)
        {
            foreach (Student student in students)
            {
                if (student.name != null)
                    Console.WriteLine($"{student.name} has score {student.score}");
                }
            }
        }

        static public void AddScores(Student[] students)
        {
            int scoreCount = 0;

            Console.WriteLine("Please enter your student scores or x to end");

            while (true)
            {
                int score = Validate();
                if (score == STOP_CODE)
                {
                    return;
                }
                else
                {
                    Console.Write("Enter the student first name: ");
                    string studentInput = Console.ReadLine();
                    students[scoreCount] = new Student(studentInput, score);
                }
                scoreCount = scoreCount + 1;
                if (scoreCount == MAX_STUDENTS)
                {
                    Console.WriteLine("You cannot enter any more students");
                    return;
                }
            }
        }
        static public int getScore(Student student)
        {
            return Convert.ToInt32(student.score);
        }

        static public int FindScore(string studentName, Student[] students)
        {
            foreach (Student student in students)
            {
                if (student.name == studentName)
                {
                    return student.score;
                }
            }
            return STOP_CODE;
        }

        // Main program
        static void Main(string[] args)
        {
            Console.WriteLine("Welcome to AQA Student Scores");

            while (true)
            {
                Console.WriteLine("");
                Console.WriteLine("#################################################");
                Console.WriteLine("Please enter a menu choice");
                Console.WriteLine("Enter 1 to enter student scores");
                Console.WriteLine("Enter 2 to sort and print student scores");
                Console.WriteLine("Enter 3 to find a student's score");
                Console.WriteLine("Enter 4 to quit the program");
                Console.WriteLine("#################################################");
                Console.Write("Menu choice: ");
                string menuItem = Console.ReadLine();
                switch (menuItem)
                {
                    case "1":
                        Console.WriteLine("Add scores");
                        AddScores(students);
                        Console.WriteLine("Added students: ");
                        PrintStudents(students);
                        break;
                    case "2":
                        students = students.OrderBy(Student => Student.score).ToArray();
                        Console.WriteLine("Sorted students");
                        PrintStudents(students);
                        break;
                    case "3":
                        Console.Write("Enter student name to find: ");
                        string studentName = Console.ReadLine();
                        int studentScore = FindScore(studentName, students);
                        if (studentScore != STOP_CODE)
                        {
                            Console.Write($"Student {studentName} has a score of {studentScore}");
                        }
                        else
                        {
                            Console.WriteLine($"{studentName} not found");
                        }
                        break;
                    case "4":
                        return;
                    default:
                        Console.WriteLine("Unknown option selected!");
                        break;
```

```
                                        }
                              }
                    }
          }
}
```