Examiners' Report
Lead Examiner Feedback

June 2022

Pearson BTEC Nationals
In Computing (31771H)
Unit 4: Software Design and Development Project

Edexcel and BTEC Qualifications

Edexcel and BTEC qualifications come from Pearson, the world's leading learning company. We provide a wide range of qualifications including academic, vocational, occupational and specific programmes for employers. For further information visit our qualifications website at http://qualifications.pearson.com/en/home.html for our BTEC qualifications.

Alternatively, you can get in touch with us using the details on our contact us page at http://qualifications.pearson.com/en/contact-us.html

If you have any subject specific questions about this specification that require the help of a subject specialist, you can speak directly to the subject team at Pearson. Their contact details can be found on this link: http://qualifications.pearson.com/en/support/support-for-you/teachers.html

You can also use our online Ask the Expert service at https://www.edexcelonline.com
You will need an Edexcel Online username and password to access this service.

Pearson: helping people progress, everywhere

Our aim is to help everyone progress in their lives through education. We believe in every kind of learning, for all kinds of people, wherever they are in the world. We've been involved in education for over 150 years, and by working across 70 countries, in 100 languages, we have built an international reputation for our commitment to high standards and raising achievement through innovation in education. Find out more about how we can help you and your learners at: www.pearson.com/uk

June 2022
Publications Code 31771H_2206_ER

Grade Boundaries

What is a grade boundary?

A grade boundary is where we set the level of achievement required to obtain a certain grade for the externally assessed unit. We set grade boundaries for each grade, at Distinction, Merit and Pass.

Setting grade boundaries

When we set grade boundaries, we look at the performance of every learner who took the external assessment. When we can see the full picture of performance, our experts are then able to decide where best to place the grade boundaries – this means that they decide what the lowest possible mark is for a particular grade.

When our experts set the grade boundaries, they make sure that learners receive grades which reflect their ability. Awarding grade boundaries is conducted to ensure learners achieve the grade they deserve to achieve, irrespective of variation in the external assessment.

Variations in external assessments

Each external assessment we set asks different questions and may assess different parts of the unit content outlined in the specification. It would be unfair to learners if we set the same grade boundaries for each assessment, because then it would not take accessibility into account.

Grade boundaries for this, and all other papers, are on the website via this link

Awarding BTEC qualifications in 2022

Ofqual has set out their plans for awarding qualifications in 2022 and intend to return to a normal, pre-pandemic, approach to grading standards over by 2023. They have confirmed that 2022 will be a transition year, to reflect that we are in a pandemic recovery period and students' education has been disrupted.

Our guiding principle and approach to awarding BTEC qualification results in 2022 will be to ensure parity in relation to the approach being taken for GCSE and A level learners. BTEC courses have a different structure and design to academic qualifications - BTECs are modular qualifications (with assessments taking place throughout the course) compared to GCSEs and A levels which

are linear (assessed and awarded at the same time at the end of the year), and therefore our approach needs to be different.

In 2022 we will return to the usual method of calculating BTEC qualification results, however adaptations including, U-TAGs and reduced internal assessment, are in place to provide a comprehensive package of support for students.

The basis of our awarding approach to BTECs this year is to ensure it is as fair as possible for all learners. We will use a range of evidence to set grade boundaries for the external units. Part of this evidence will be to closely monitor learner performance in all assessments that contribute to learners' final qualification grade, to ensure parity with A level and GCSEs.

Further information can be found on our website and via our Social Media channels.

Introduction

This was the seventh examination series for Level 3 BTEC Computing Unit 4: Software Design and Development Project.

This unit is a paper-based exam, assessed through a task-based assessment. The set task assesses learners' ability to design, create and evaluate software using Python (3.4 or a later version) or one of the C family programming languages. This unit is a mandatory unit for all learners studying the extended diploma.

The examination for this unit will always contain five activities and each one will be linked to a scenario. The scenario is clearly stated at the beginning of each assessment. The activities will test learners on different areas of the specification, and learners are expected to apply their knowledge to the scenario.

All Activities of the examination paper provide differentiation at all attainment levels and the brief is designed to escalate in difficulty so that a larger percentage of higher-grade marks depends on the skills, knowledge, understanding and application of theory.

Introduction to the Overall Performance of the Unit

The overall performance of this unit during the Summer 2022 session is comparable to the previous summer series (Summer 2022).

The performance on Activity 1 resulted in most learners picking up marks in band 2. Majority of the responses used BCS symbols, the better responses were able to break down the requirements into relevant parts. Learners provided evidence of links between component parts but evidence of handling errors within the flowcharts was not always present. In many instances, learners tried to fit the entire flowchart in one page, which often made them difficult to read and follow. Many learners applied validation, mostly to password length. Although some learners used incorrect logic at the boundaries, resulting in the wrong password length being accepted or rejected. Another common error was to reverse the "yes" "no" labels on the decision box.

Activity 2 was generally completed to a good standard, where learners demonstrated their ability to apply pseudocode design methodologies to the scenario. Some learners produced pseudocode that was very similar to the code produced in activity 4. This reduces clarity and readability for anyone not familiar with the language specific functions or syntax. The aim of the pseudocode is to provide a step in the design process that would allow a third party if needed to continue with the coding and any of the specified languages.

Activity 3 & 4 (testing) is still an area of concern, where learners tend to underperform. Many learners are only demonstrating skills to access mark band level 1. It is recommended that centres reinforce to learners what a test plan should consist of, and the importance of testing throughout the whole design and development which is essential for accessing higher marks. Learners mostly tested the inputs, with some limited data, stating simply that the data would be accepted / rejected. However, most learners did not test the final calculations using an example password and manually calculated scores and ratings. For example, learners frequently stated the test data was for a “very high” rated password, but did not state the actual password being used as the test data.

Activity 4 (Coding) was well completed by some learners with marks awarded in the top mark band as they produced a working solution along with detailed comments. These learners validated the inputs well and in trapped errors due to wrong data types being input by the user. In addition to checking the password length, some learners tested for invalid special characters and rejected the password prior to starting the scoring section of the algorithm. A common issue with validation is still lack of feedback to user indicating why the input had been rejected. Meaningful error messages would improve the code and the user experience.

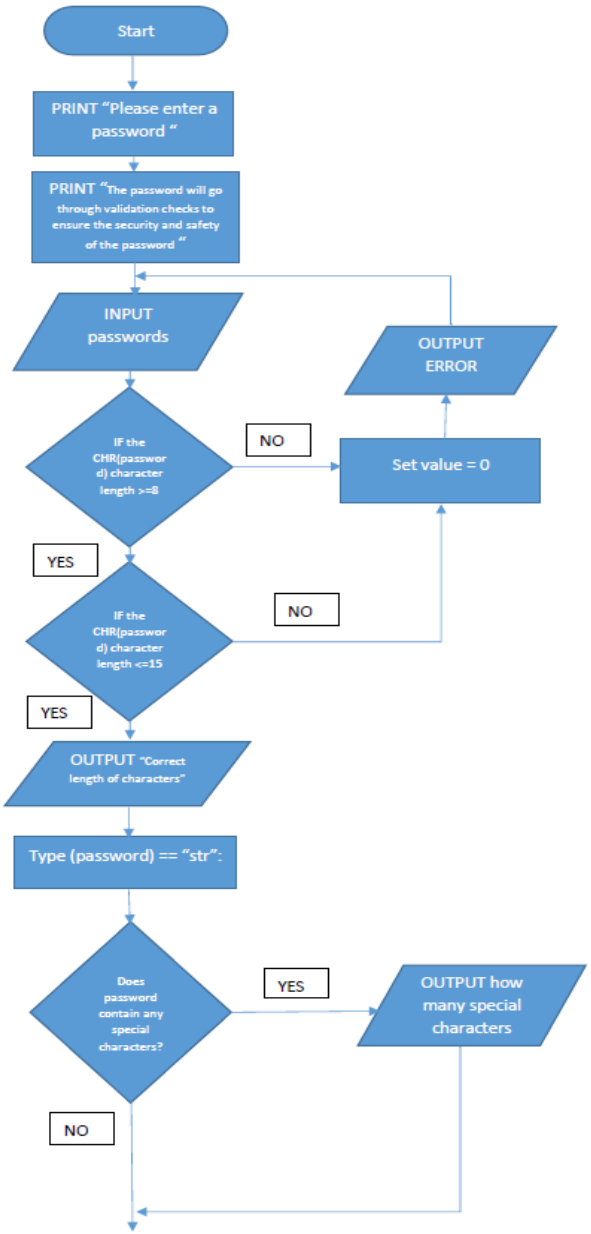
The evaluations (activity 5) were of a good standard and most learner’s accessed bands two and three. Some learners only produced a step-by-step account of what they did which resulted in marks from band 1 being awarded.

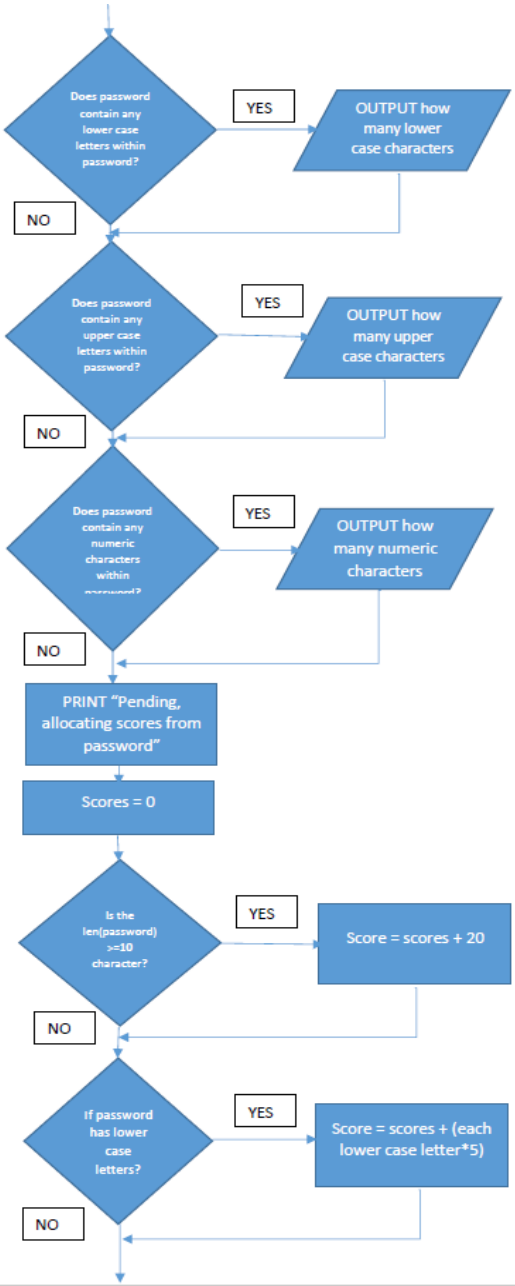
Individual Questions

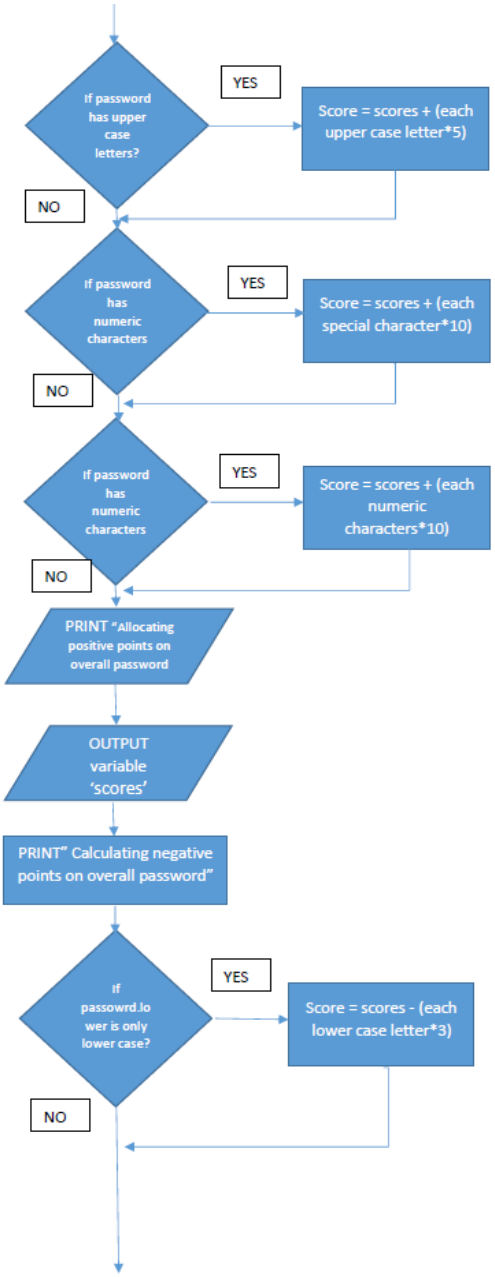
The following section considers each question on the paper, providing examples of learner responses and a brief commentary of why the responses gained the marks they did. This section should be considered with the live external assessment and the corresponding mark scheme.

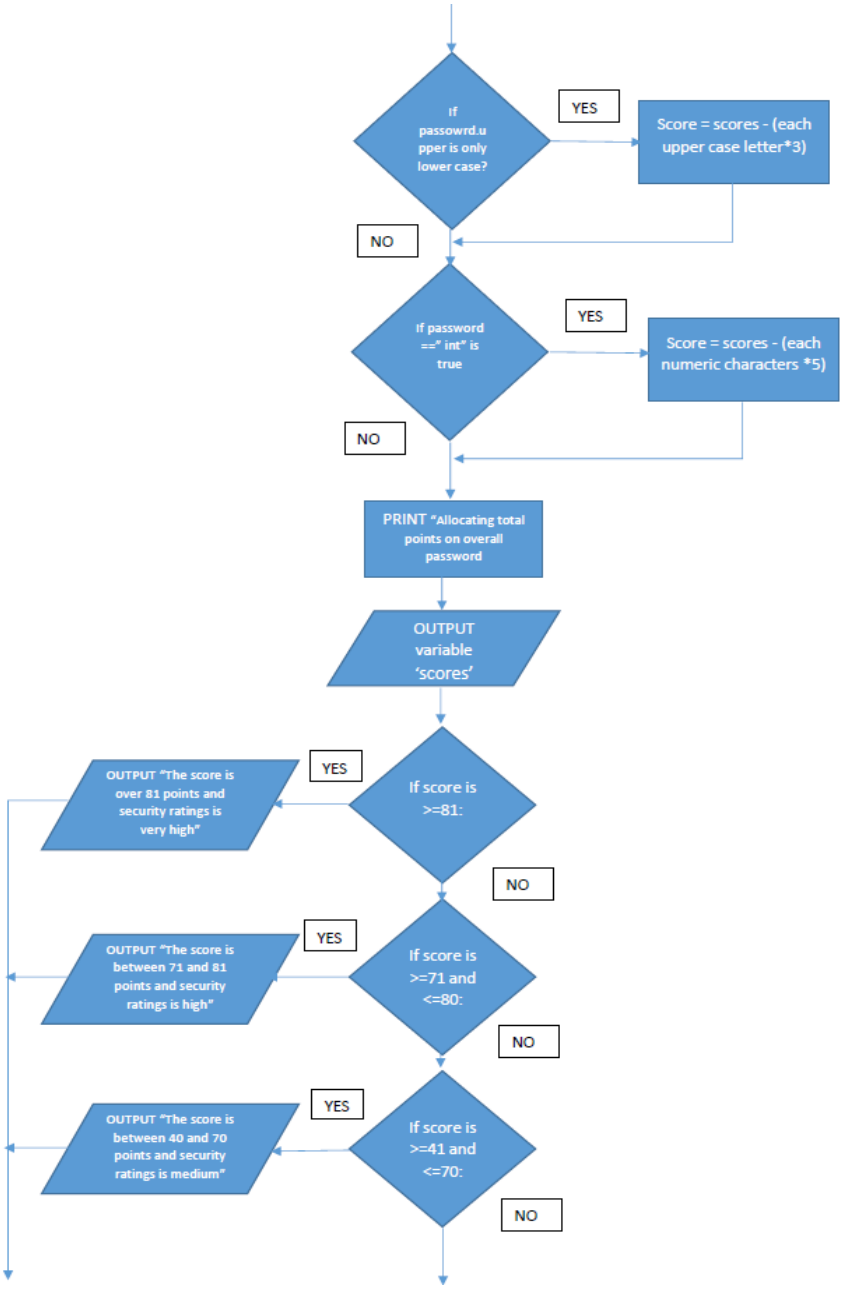
Activity 1

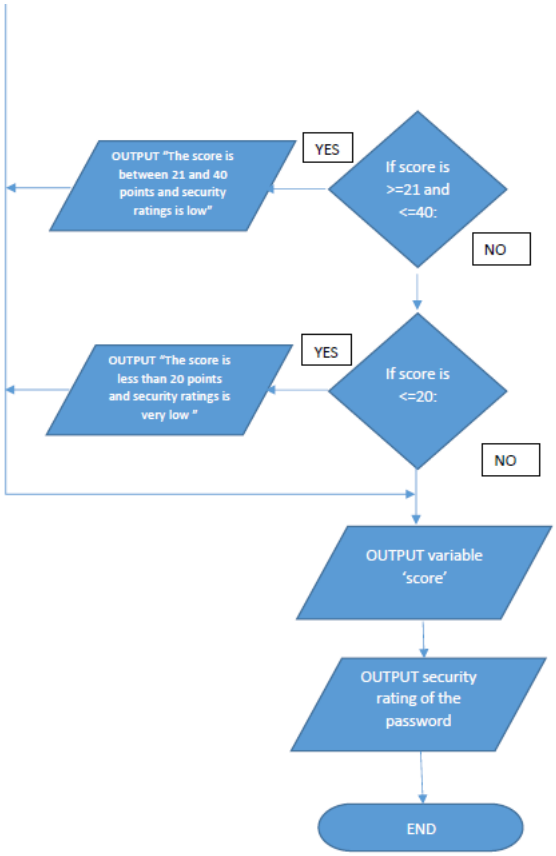
Example 1:




Does password contain any lower case letters within password?
YES
OUTPUT how many lower case characters
NO
Does password contain any upper case letters within password?
YES
OUTPUT how many upper case characters
NO
Does password contain any numeric characters within
YES
OUTPUT how many numeric characters
NO
PRINT "Pending, allocating scores from password"
Scores = 0
Is the len(password) >=10 character?
YES
Score = scores + 20
NO
If password has lower case letters?
YES
Score = scores + (each lower case letter*5)
NO


If password has upper case letters?
YES
Score = scores + (each upper case letter*5)
NO
If password has numeric characters
YES
Score = scores + (each special character*10)
NO
If password has numeric characters
YES
Score = scores + (each numeric characters*10)
NO
PRINT "Allocating positive points on overall password
OUTPUT variable 'scores'
PRINT" Calculating negative points on overall password"
If passowrd.lower is only lower case?
YES
Score = scores - (each lower case letter*3)
NO


If passowrd.upper is only lower case?
YES
Score = scores - (each upper case letter*3)
NO
If password ==" int" is true
YES
Score = scores - (each numeric characters *5)
NO
PRINT "Allocating total points on overall password
OUTPUT variable 'scores'
If score is >=81:
YES
OUTPUT "The score is over 81 points and security ratings is very high"
NO
If score is >=71 and <=80:
YES
OUTPUT "The score is between 71 and 81 points and security ratings is high"
NO
If score is >=41 and <=70:
YES
OUTPUT "The score is between 40 and 70 points and security ratings is medium"
NO

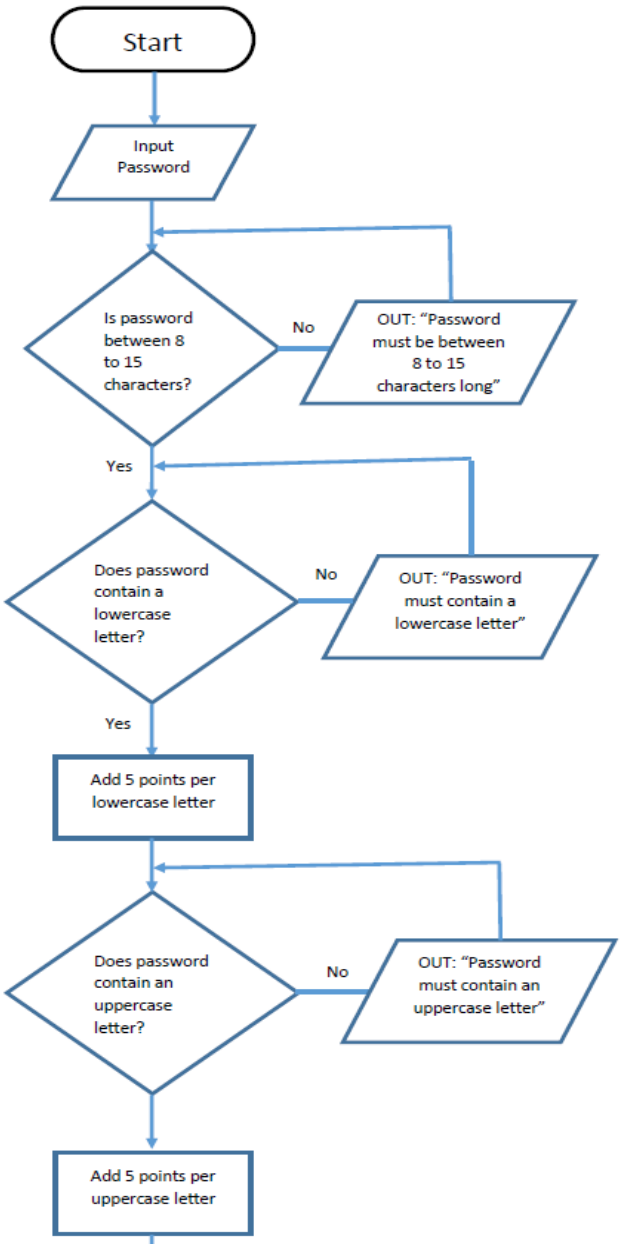


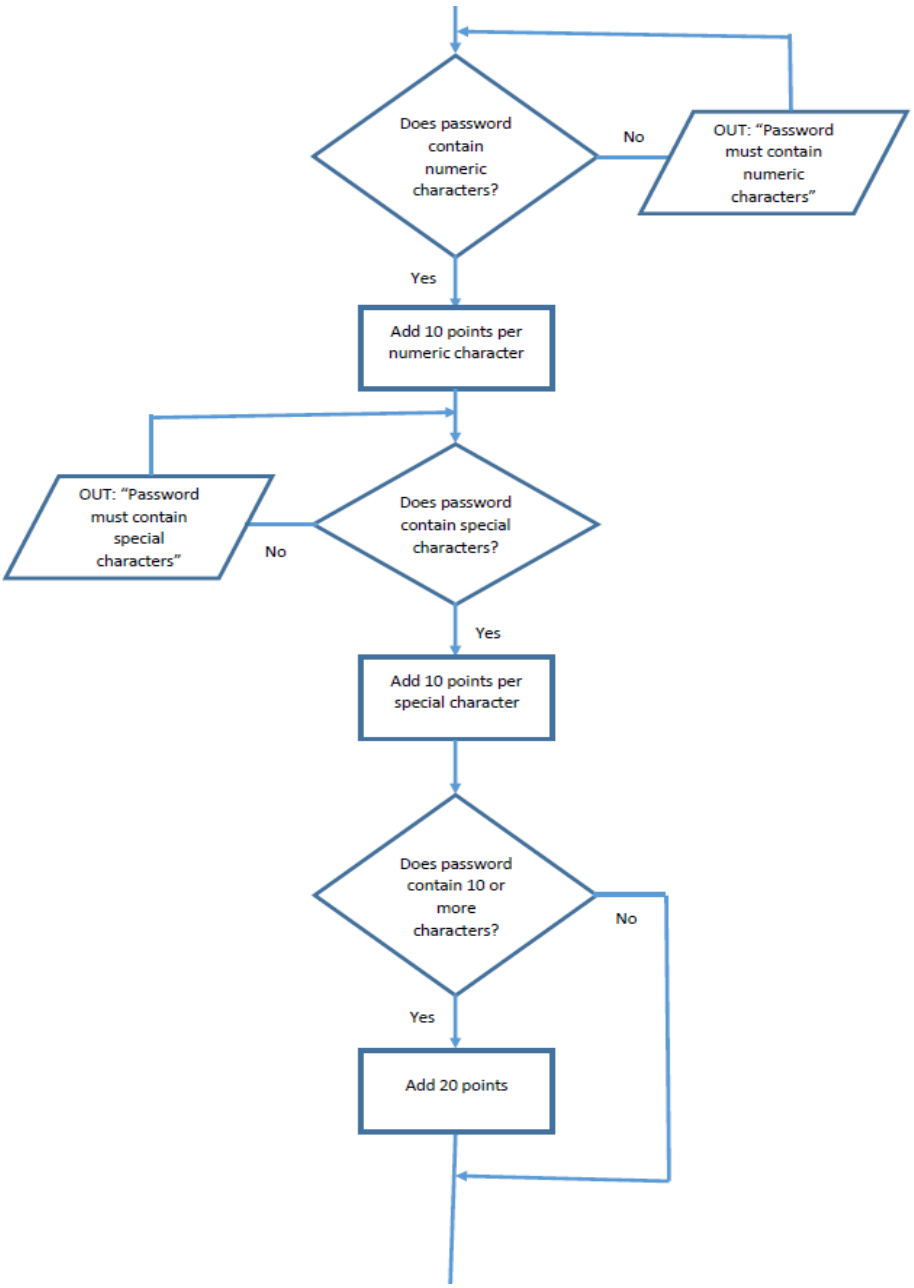
Lead examiner commentary:

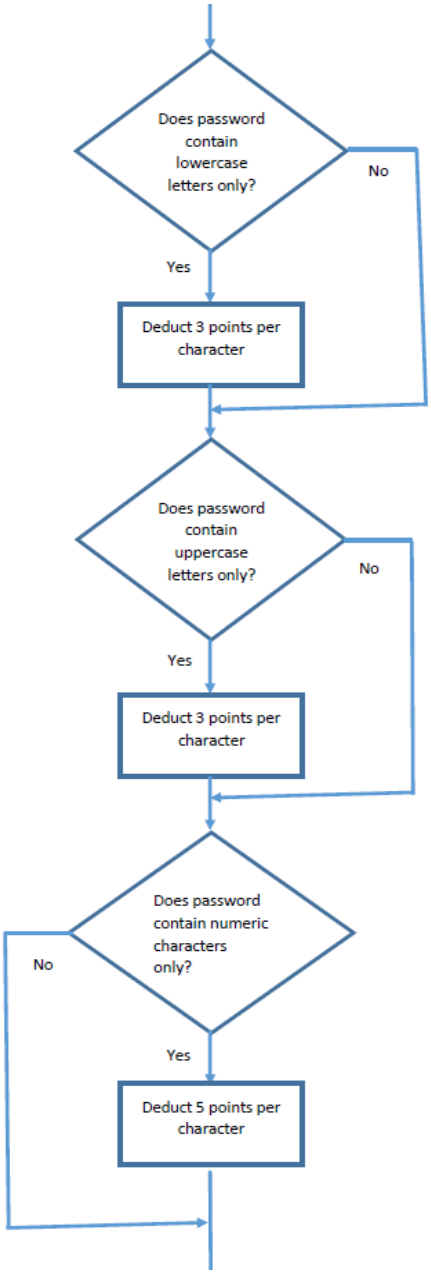
The learner shows accurate use of BCS symbols. The logic is correct, the flowchart is easy to follow and breaks down requirements into component parts. Shows full coverage of input and outputs, the naming conventions used are appropriate and consistent.

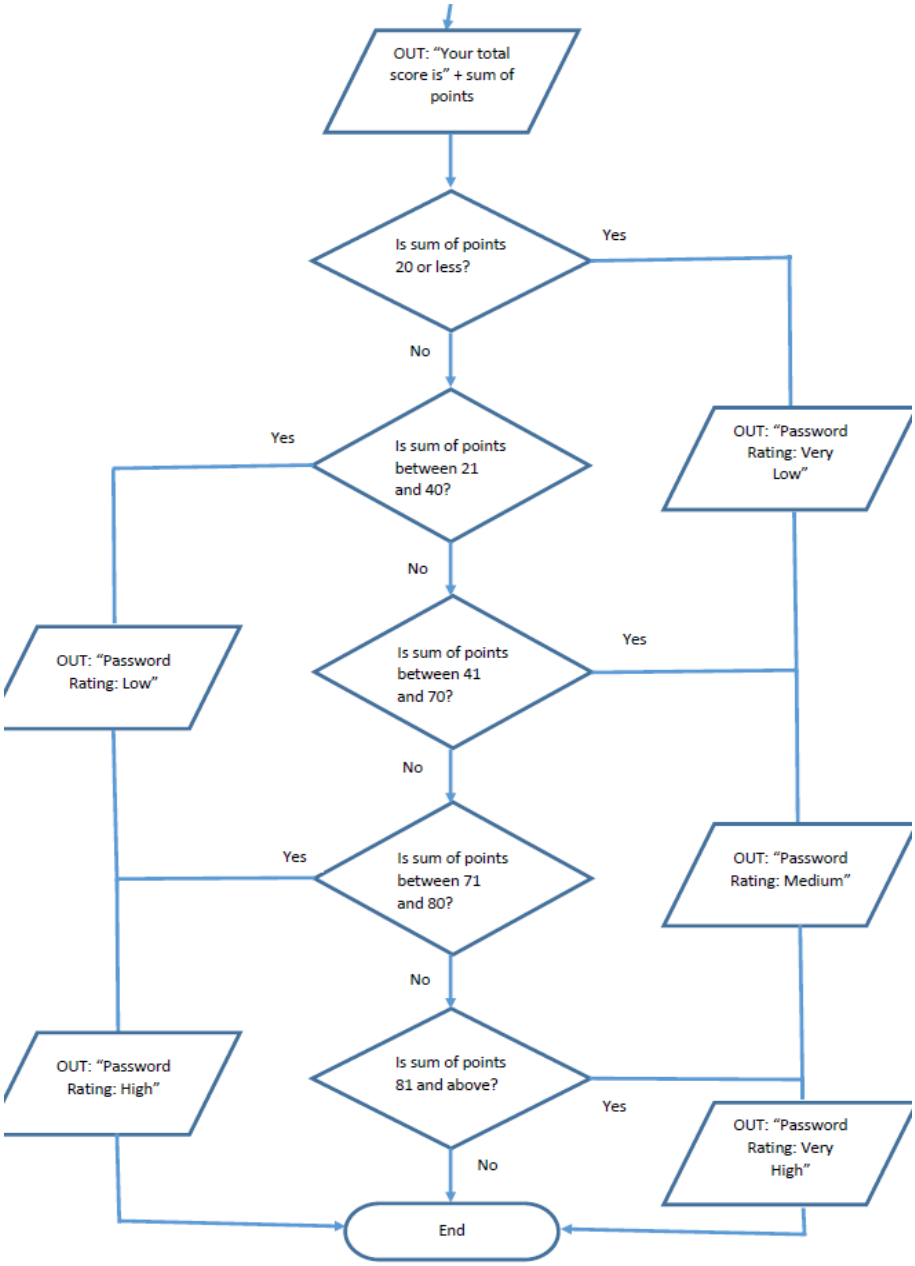
Band 3 (8 marks).

Example 2:


Start
Input Password
Is password between 8 to 15 characters?
No
OUT: "Password must be between 8 to 15 characters long"
Yes
Does password contain a lowercase letter?
No
OUT: "Password must contain a lowercase letter"
Yes
Add 5 points per lowercase letter
Does password contain an uppercase letter?
No
OUT: "Password must contain an uppercase letter"
Add 5 points per uppercase letter


Does password contain numeric characters?
No
OUT: "Password must contain numeric characters"
Yes
Add 10 points per numeric character
OUT: "Password must contain special characters"
No
Does password contain special characters?
Yes
Add 10 points per special character
Does password contain 10 or more characters?
No
Yes
Add 20 points


Does password contain lowercase letters only?
No
Yes
Deduct 3 points per character
Does password contain uppercase letters only?
No
Yes
Deduct 3 points per character
Does password contain numeric characters only?
No
Yes
Deduct 5 points per character



Lead examiner commentary:

The learner shows use of BCS symbols, coverage of inputs, the learner is inconsistent with naming conventions. The logic for the password rating is difficult to follow as depending on the points score one or messages would be produced. For example, following the yes line from the 20 points or less decision, would if the flowchart is followed produce 3 messages, as the line is continuous down to the end of the flowchart.

Mark in **band 1 (3 marks)**.

Activity 2

Example 1:

```
BEGIN
SET finalScore to 0
SET numScore  to 0
SET lowerScore to 0
SET upperScore to 0
SET spclScore to 0
SET spclChar to !,%,&,*,+,=

DISPLAY "Please enter your password"
SET validationLoop to True
  SET usrPas to input
  IF the length of usrPas is <8 or >15
        DISPLAY "Characters must be between 8 and 15"
        DISPLAY "Please re-enter your password within the character limit"
        END
  ELSE:
        IF usrPas is all numeric
              FOR every character in usrPas
              SET finalScore as -=5
        IF usrPas is all uppercase
              FOR every character in usrPas
              SET finalScore as -=3
        IF usrPas is all lowercase
              FOR every character in usrPas
              SET finalScore as -=3
        SET validationLoop to False

FOR every character in usrPas
        IF there is a character from spclChar
              SET finalScore as +=10
              SET spclScore as +=10
        ELIF if there is a character that is numeric
              SET finalScore +=10
              SET numScore +=10
        ELIF if there is a character that is uppercase
              SET finalScore +=5
              SET upperScore +=5
        ELIF if there is a character that is lowercase
              SET finalScore +=5
              SET lowerScore +=5
        ELSE
              DISPLAY ("You have entered an invalid character, please re-enter your password
              with valid characters. Your allowed list of special characters are,
              DISPLAY spclChar
              END

IF finalScore is >20
        Output "Your rating is very low, you have a score of"
```

```
        Output finalScore
ELIF finalScore is >20 and <=40
        Output "Your rating is low, you have a score of"
        Output finalScore
ELIF finalScore is >40 and <=70
        Output "Your rating is medium, you have a score of"
        Output finalScore
ELIF finalScore is >70 and <=80
        Output "Your rating is high, you have a score of"
        Output finalScore
ELSE
        Output "Your rating is very high, you have a score of"
        Output finalScore

Output "Your numeric rating is:"
Output numScore

Output "Your uppercase rating is:"
Output upperScore

Output "Your lowercase rating is:"
Output lowerScore

Output "Your special character rating is:"
Output spclScore
```

Lead examiner commentary:

The learner has produced a structure which shows appropriate and consistent use of hierarchy and indentation, providing clarity and mostly readable pseudocode. The pseudocode will provide a working solution with some minor errors. Appropriate naming conventions have been used and precise use of logical operations.

Mark in **band 3 (8 marks)**.

Example 2:

```
START
Points = 0
INPUT Password ("What is your password")
IF password is <8 or >15 then
        OUTPUT ("Invalid Password")
        BREAK
ELSE
NumOfChars=Password(len)
Check Password is all upper or lower or numeric
IF Password = upper
        FOR NumOfChars in Password
                Points -3
        END
ELIF Password = lower
        FOR NumOfChars in Password
                Points -3
        END
ELIF Password = Int
        FOR NumOfChars in Password
                Points -5
        END
ELSE
        FOR Lower in Password
                Points +5
        END
        FOR Upper in Password
                Points +5
        END
        FOR Int in Password
                Points +10
        END
```

```
        FOR Special in Password
                Points +10
        END
IF password len >10
        Points +20
ELSE
IF points =<20
        OUTPUT ("Your password rating is Very Low" +Points)
ELIF points =>21 OR =<40
        OUTPUT ("Your password rating is Low" +Points)
ELIF points =>41 OR =<70
        OUTPUT ("Your password rating is Medium" +Points)
ELIF points =>71 OR =<80
        OUTPUT ("Your password rating is High"+Points)
ELSE
        OUTPUT ("Your password rating is Very High" +Points)
END
```

Lead examiner commentary:

The learner has produced a structure that shows some appropriate hierarchies, readability is limited, and the detail of the calculations in the scoring section is very limited. The code uses appropriate naming conventions but there are inconsistencies. There is some imprecise use of logical operations, leading to an incomplete solution.

Mark in **band 1 (3 marks)**.

Activity 3

Example 1:

Test Number	Purpose of test	Test Data	Expected Result
1	Testing to see if the program will accept a password less than 8 characters	"B"	Error output due to the length not meeting the criteria
2	Testing to see if the program will accept a password over 15 characters	"password1234qwen"	Error output due to the length extending over the criteria
3	Testing to see if the program will accept a password that's exactly 8 characters	"Password"	Password should be accepted and the program should continue
4	Testing to see if the program will accept a password that's exactly 15 characters	"Passwordpasswor"	The program should accept the password entered and continue
5	Checking if the program checks if the password only contains lower case characters	"password"	An output should state that the user has been deducted 3 points per character due to the password only containing one type of character
6	Checking if the program checks if the password only contains upper case characters	"PASSWORD"	An output should state that the user has been deducted 3 points per character due to the password only containing one type of character
7	Checking if the program checks if the password only contains numeric characters	"12345678"	An output should state that the user has been deducted 5 points per character due to the password only containing one type of character
8	Testing to see if the points are allocated to each character accordingly	"Password"	An output should state that the password is worth 40 points due to there being 1 upper and 7 lower
9	Testing to see if the points are allocated to each character accordingly	"PaSsWoRd"	An output should state that the password is worth 40 points due to there being 4 upper and 4 lower
10	Testing to see if the points are allocated to each character accordingly	"Passw0rd"	An output should state that the password is worth 45 due to there being 1 upper, 6 lower and 1 numeric character
11	Testing to see if the points are allocated to each character accordingly	"Passw0rd!"	An output should state that the password is worth 55 points due to there being 1 upper, 6 lower, 1 numeric and 1 special character
12	Testing to see if the points are allocated to each character accordingly	"Password12"	An output should state that the password is worth 80 points due to there being 1 upper and 7 lower and 2 numeric characters, as well as the password meeting the criteria of containing 10 or more characters the password is awarded an extra 20 points
13	Testing the output depending on the points awarded for the password	20 points or less	An output should state that the users password security rating is very low

14	Testing the output depending on the points awarded for the password	21 to 40 points	An output should state that the users password security rating is low	
15	Testing the output depending on the points awarded for the password	41 to 70 points	An output should state that the users password security rating is medium	
16	Testing the output depending on the points awarded for the password	71 to 80 points	An output should state that the users password security rating is high	
17	Testing the output depending on the points awarded for the password	81 points and above	An output should state that the users password security rating is very high	
18	If no password is entered	" "	An error should display telling the user to enter a password	

Lead examiner commentary:

The learner has produced a thorough test plan to confirm a working solution which includes a range of data. Expected results are specific and accurate based on identified test data. The examples of passwords are given in the test data and the expected results are described in some detail. The learner is testing the security rating, however lacks the actual passwords used for the tests and simply gives the points range used, therefore there is not enough detail achieve mark band 3

Mark in **band 2 (4 marks)**.

Example 2:

Document for Activities 3 and 4

Test Plan (add additional rows as required)

Program language the product is to be produced in (tick box for language used)

Python ☐ C Family ☐

Test Number	Purpose of test	Test Data	Expected Result
1	Can you input your password	Testing if the first input is working as intended	We expect that you can input your password and the scoring system should begin
2	Can the password length be checked	Checking the IF statement can check if the password is the correct length	The password length should be checked and the program should continue as intended
3	Can the program identify if the password is all upper case, lowercase or numerical	Checking to see if the program can identify and deduct points accordingly	I expect the functions to work and we have no errors
4	The program should output your points total and rating	Testing my program is functional and can display data	The user gets a message display showing their rating and points
5	Checking the program is functional from start to finish	The final test which will test the overall quality of the program as I try to break it	The program should not be able to broken or exploited and errors should not occur

Lead examiner commentary:

The plan describes testing of inputs, the is no actual test data given just a description of the section of code being tested. There is some testing of the logic described. The plan is too narrow to confirm a working solution

.

Mark in **band 1 (1 mark)**.

Activity 4 Code

Example 1:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace exam_code
{
   class Program
   {
      static void Main(string[] args)
      {
         /*I am using two strings a string named "password", this string will save the user
input.
          i am also using a  string named "SpecialCharacters", this string contains all of
the speical characters allowed
          in the password, my for loop will access this later.
         */
         /* I have declared multiple ints that will work as my counter, for example my for
loop will count how many numbers are
           in the password and save the number to "numbers".
          */
         string password;
         string specialCharacters = "!,%,&,*,+,=";
         int upperCase = 0, lowerCase = 0, numbers = 0, specialChar = 0, negative = 0,
score = 0, totalScore = 0;
         Console.WriteLine("Please enter a password between 8 and 15 characters
long");
         password = Console.ReadLine();
         // this while loop will loop until the user meets the required password
spesification
         while (password.Length < 8 || password.Length > 15)
         {
            if (password.Length < 8)
            {
               Console.WriteLine("Your password is too short \n Please try again");
               password = Console.ReadLine();
            }
            else if (password.Length > 15)
            {
               Console.WriteLine("Your password is too long \n Please try again");
               password = Console.ReadLine();
            }
         }
         /* This for loop will check through the users password and check for lowercase,
uppercase, numeric and
          special characters, the specific varable will increase when a character is found.
          The loop also increase the score by each characters value.
          */
```

```
            for (int pos = 0; pos < password.Length; pos++)

            {
                char ch = password[pos];
                //saving password[pos] as a char saves time as i dont have to write
password[pos] every time.
                if (ch >= 'a' && ch <= 'z')
                {
                    lowerCase++;
                    score += 5;
                }
                else if (ch >= 'A' && ch <= 'Z')
                {
                    upperCase++;
                    score += 5;
                }
                else if (ch >= '0' && ch <= '9')
                {
                    numbers++;
                    score += 10;
                }
                else if (specialCharacters.Contains(ch))
                {
                    specialChar++;
                    score += 10;
                }

            }
            Console.Clear();
            //I used console.claer to remove and unnessary text from the console
            if (password.Length >= 10)
            {
                score += 20;
                Console.WriteLine("Your password contained more than 10 characters +20
points added to score");
            }
            /* this if statment checks to see if the user has used only one type of character
             for example lowercase characters, if this is the case the users score will be
deducted
             and the console will produce "Your password only consists of lowercase
characters"
             */
            if (password.Length == lowerCase || password.Length == upperCase)
            {
                negative = password.Length * 3;
                Console.WriteLine("Your password only consists of lower or upper case
characters");

            }

            else if (password.Length == numbers)
            {
                negative = password.Length * 5;
                Console.WriteLine("Your password only consists of numerical characters");
            }
```

```
            //the total score is worked out by taking the negative int value away from the
score int
            totalScore = score - negative;

            /*This if statment uses the "totalScore" int to determin the password rating
             for example if the total score is less than or equal to 20 the console will
produce "Password is rated Very low".
            */


            if (totalScore <= 20)
            {
                Console.WriteLine("Password is rated Very low");
            }
            else if (totalScore >= 21 && totalScore <= 40)
            {
                Console.WriteLine("Password is rated Low");
            }
            else if (totalScore >= 41 && totalScore <= 70)
            {
                Console.WriteLine("Password is rated Medium");
            }
            else if (totalScore >= 71 && totalScore <= 80)
            {
                Console.WriteLine("Password is rated High");
            }
            else if (totalScore >= 81)
            {
                Console.WriteLine("Password is rated Very High");
            }
            /* This is where the users password data is printed to the console, i used
Console.WriteLine with $ to print a message
             alongside the users data for example "Your password contained {lowerCase}
lowercase characters would print
             "Your password contained 6 lowercase characters
            */
            Console.WriteLine($"Your password score is {totalScore}");
            Console.WriteLine($"Your password contained {lowerCase} lowercase
characters\nLowercase characters scored {lowerCase * 5} points");
            Console.WriteLine($"Your password contained {upperCase} uppercase
characters \nUppercase characters scored {upperCase * 5} points");
            Console.WriteLine($"Your password contained {numbers} numbers \nNumerical
characters scored {numbers * 10} ");
            Console.WriteLine($"Your password contained {specialChar} special characters
\nSpecial characters scored {specialChar * 10} points");
            Console.WriteLine($"Points deducted {negative}");
            // I used Console.ReadKey to keep stop the console from closing
            Console.ReadKey();
        }


    }
}
```

Lead examiner commentary:

The learner has produced a program that fully meets all the requirements. Accurate syntax and indentation have been used throughout the code and commenting is consistently clear and informative. Program outputs are accurate and informative, validation and other checks have been used which are all accurate, resulting in a largely robust program being created.

Mark in **band 4 (24 marks)**.

Example 2:

```
Points=0 #Declaring the points variable and setting is as zero
number=["1","2","3","4","5","6","7","8","9","0"] #Array for numericals that we use later to validate the password
special=["<",">","?","/",".","~","@",":",";","{","}","=","-","_","+"]
Password=input(str("What is your password: "))
#Decarling the password variable and letting the user input their password and #defining it as a string

if len(Password) <8:          #The len here will check the length of the input and count how many characters are stored within it
    print("Your password is too short")   #If statement to check if the password is shorter than 8 characters
elif len(Password) >15:
    print("Your password is too long")    #Elif statement to check if the password is longer than 15 characters
else:
    print("Your password is the correct length") #Ensuring the user that their password meets the required length


if Password.isupper():
    print("Your password is all uppercase that is a -3 point deduction per character") #Checking if the password is all upper and deducting points
    deduction=len(Password)-3   #Getting the length of the password again to deduct points
    Points=Points-deduction
elif Password.lower():
    print("Your password is all lowercase that is a -3 point deduction per character") #Checking if the password is all lower and deducting points
    deduction2=len(Password)-3
    Points=Points-deduction2
elif Password == number:
    print("Your password is all numerical that is a -3 point deduction per character") #Checking if the password is all lower and deducting points
    deduction3=len(Password)-3
    Points=Points-deduction3

if len(Password) >10:
    print("Your password is more than 10 characters that is +20 points") #Checking the password length once again to see if it qualifies for 20 more points
    Points=Points+20

print("You will now have a +5 per uppercase or lowercase letter")
PasswordLength=len(Password)  #Getting the length of the password and defining the int as a variable
Points2=PasswordLength*5      #Calculating the points by using another points variable
Points=Points+Points2       #and adding it with the original points variable to

if Points <20:
    print("Your password rating is very low with a score of ",(Points)) #Now we display our points and rating based on the points total
elif Points >21 and Points <40:
```

```
    print("Your password rating is low with a score of ",(Points)) #Using elif to help
display users ratings correctly by saying "if points are more than 21" then we print the
rating and score
elif Points >41 and Points <70:
    print("Your password rating is medium with a score of ",(Points))
elif Points >71 and Points <80:
    print("Your password rating is high with a score of ",(Points))
elif Points >81:
    print("Your password rating is very high with a score of ",(Points))
```

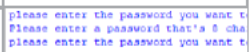
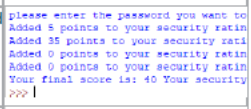
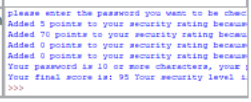
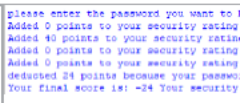
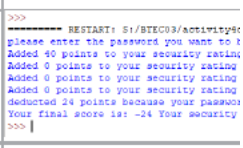
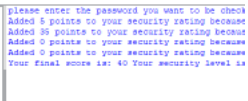
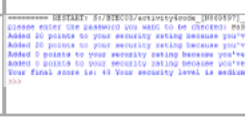
Lead examiner commentary:

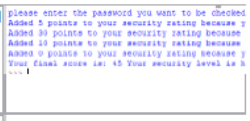
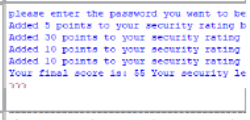
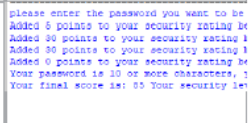
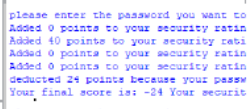
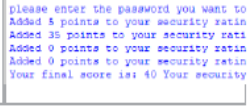
The learner has produced code that uses correct syntax. There are some logic errors in the code that mean the results are not accurate. For example the section validating the password length does not use any iteration to force the user to correct the length. The code tests the length, produces messages describing the error, however it then continues to run the next section of code.

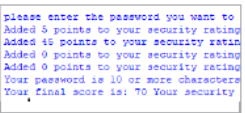
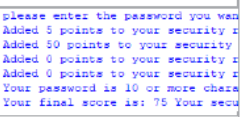
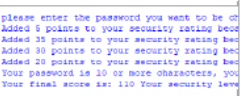
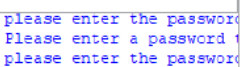
Mark in **band 2 (7 marks).**

Activity 4 Testing

Example 1:

Test Number	Purpose of test	Test Data	Expected Result	Actual Result	Comments
1	Testing to see if the program will accept a password less than 8 characters	"B"	Error output due to the length not meeting the criteria	The program posts an error message informing the user that they need to make their password 8 characters or more	
2	Testing to see if the program will accept a password over 15 characters	"password1234qwen"	Error output due to the length extending over the criteria	The program posts an error stating that the password entered doesn't match the criteria set and they need to make a password that's 15 characters or less	
3	Testing to see if the program will accept a password that's exactly 8 characters	"Password"	Password should be accepted and the program should continue	The program accepts the given password and continues through the rest of the program	
4	Testing to see if the program will accept a password that's exactly 15 characters	"Passwordpasswor"	The program should accept the password entered and continue	The program accepts the password given and continues	
5	Checking if the program checks if the password only contains lower case characters	"password"	An output should state that the user has been deducted 3 points per character due to the password only containing one type of character	The program detects the password is all lower case and deducts the points from the password accordingly	
6	Checking if the program checks if the password only contains upper case characters	"PASSWORD"	An output should state that the user has been deducted 3 points per character due to the password only containing one type of character	The program detects the password is upper case and deducts points	
7	Checking if the program checks if the password only contains numeric characters	"12345678"	An output should state that the user has been deducted 5 points per character due to the password only containing one type of character		
8	Testing to see if the points are allocated to each character accordingly	"Password"	An output should state that the password is worth 40 points due to there being 1 upper and 7 lower	Points are assigned correctly to the password	
9	Testing to see if the points are allocated to each character accordingly	"PaSsWoRd"	An output should state that the password is worth 40 points due to there being 4 upper and 4 lower	Points are assigned correctly to the given password	

10	Testing to see if the points are allocated to each character accordingly	"Passw0rd"	An output should state that the password is worth 45 due to there being 1 upper, 6 lower and 1 numeric character	Points are assigned correctly to all characters in the given password	
11	Testing to see if the points are allocated to each character accordingly	"Passw0rd!"	An output should state that the password is worth 55 points due to there being 1 upper, 6 lower, 1 numeric and 1 special character	The password is allocated the correct amount of points	
12	Testing to see if the points are allocated to each character accordingly	"Password12"	An output should state that the password is worth 80 points due to there being 1 upper and 7 lower and 2 numeric characters, as well as the password meeting the criteria of containing 10 or more characters the password is awarded an extra 20 points	The password assigns the points to each character correctly giving the desired point total	
13	Testing the output depending on the points awarded for the password	20 points or less	An output should state that the users password security rating is very low	The program detects the very low point level and displays the correct message	Due to the password minimum being 8 the password will default over 20, so I used a prevoiusly tested function of all lower case to test result 
14	Testing the output depending on the points awarded for the password	21 to 40 points	An output should state that the users password security rating is low	The program detects the low point level and displays the correct message	

15	Testing the output depending on the points awarded for the password	41 to 70 points	An output should state that the users password security rating is medium	The program detects the medium point level and displays the correct message	
16	Testing the output depending on the points awarded for the password	71 to 80 points	An output should state that the users password security rating is high	The program detects the high point level and displays the correct message	
17	Testing the output depending on the points awarded for the password	81 points and above	An output should state that the users password security rating is very high	The program detects the very high point level and displays the correct message	
18	If no password is entered	" "	An error should display telling the user to enter a password	The program detects that the user hasn't entered a password and displays an error message	

Lead examiner commentary:

Comments and screen shots clearly show the process of testing and the output produced, testing and is adequate to demonstrate the program works. However, the process shown is linear, with minimal identification of errors and how they were resolved, the comments show only a limited understanding of how error were found and fixed.
To achieve mark band 2 and above there must be evidence of errors and how they have been solved.

Mark in **band 1 (2 marks).**

Example 2:

Document for Activities 3 and 4

Test Plan (add additional rows as required)

Program language the product is to be produced in (tick box for language used):

Python ☐ C Family ☐

Test Number	Purpose of test	Test Data	Expected Result	Actual Result	Comments
1	Can you input your password	Testing if the first input is working as intended	We expect that you can input your password and the scoring system should begin	The password was able to be inputted and was checked	This was the first test as it is the first part of the program and is what the program runs off
2	Can the password length be checked	Checking the IF statement can check if the password is the correct length	The password length should be checked and the program should continue as intended	Invalid syntax "<=8 or >=15"	I need to find out what us causing this error and fix it so the program can work as intended. Fixed by doing the statements separately
3	Can the program identify if the password is all upper case, lowercase or numerical	Checking to see if the program can identify and deduct points accordingly	I expect the functions to work and we have no errors	The IF statement can check if the password is fully upper or lower with the .upper and .lower function	This one was frustrating during testing, it was not functionating properly all percent of the time but has now been fixed
4	The program should output your points total and rating	Testing my program is functional and can display data	The user gets a message display showing their rating and points	The rating score displays but the points total does not	This has been fixed. I was using a + instead of a comma to help connect the points to the print statement
5	Checking the program is functional from start to finish	The final test which will test the overall quality of the program as I try to break it	The program should not be able to broken or exploited and errors should not occur	I had a few errors but I have made the program as functional as I can	This test was the final test where I tried to test the validity and functionality of my program

Lead examiner commentary:

Testing shows evidence of a limited testing process, with no identification and resolution of errors. The descriptions used show limited understanding of the testing process.

Mark in **band 1 (1 marks)**.

Activity 5

Example 1:

"How my solution meets the requirements of the scenario

Throughout my coding solution I have met and exceeded the requirements of the scenario.
Firstly, I have validated the character length, after this I have a statement which checks the length of the password to add 20 points if needed, then the password is checked to see if it is all numeric, uppercase or lowercase and points will be deducted accordingly if the password is.
Then, the user's password is checked for numbers, lowercase, uppercase and special characters and points are added based on the amount of each. If they user enters invalid numbers the program will inform them of that and display the valid special characters.
Finally, the program will print the users total score as well as their individual scores for numbers, lowercase, uppercase and special characters. I made this addition as I believe it benefits the user so not only can they see in which areas their password is weak and improve it but they can also have a more detailed overview of their password's components.
Below I have detailed this further and have more clearly portrayed my solution and its intricacies.
The quality and performance of my program
Throughout my programming solution I have continuously met the needs and requirements of the scenario. Firstly, I have appropriately validated the program for both the allowed length of the password and the allowed list of special characters. I have done this, in the case of the length, by using the 'len' function to allow me to check the length of the password, after this I then used a comparison by stating if the passwords character length is less than 8 or greater than 15 the statement will print out a response to the user telling them the character length their password must conform to and to re enter their password within the character limit.

In the case of validating the special characters I have used a list which contains the allowed special characters. From here the program will run through a list of if and elif statements which will check what characters have been entered into the usrPas variable, userPas being shorthand for User Password, if the user has entered any special characters not in the variable spclChar, meaning Special Character they will receive a print statement telling them they have entered an invalid character and to re enter the password and by using + to concatenate the string it will also display the spclChar list of the allowed special characters so they users knows which special characters they can enter.
Secondly, I have used a for loop to require the user to input their password continuously if the password they have entered is below 8 characters or above 15 they will have to re-enter their password until they enter a password between 8 and 15 characters as specified in the scenario. The else statement then bails the user out of the loop and the loop is set to false.
After the loop has been set to False and the password validated the code will check if the password, usrPas, is over or equal to 10 characters if it is 20 points will be added to finalScore. Then the code willrun through a series of if statements, these statements will check if the users password is all numeric, all upper case or all lowercase using the .isnumeric, .isupper, .islower functions. If the user's password is either all numeric, lowercase or uppercase the code will use a for loop to run through the entered password and count the characters to deduct the respective points from the finalScore variable. Deducting 5 points per character for an all numeric password and 3 points per character for an all uppercase or lowercase password.
Once the users password has been checked for point deductions the code then moves on to using a for loop, the for loop will run through every character in the password and using like above, .isnumeric, .isupper and .islower. The first part of the for loop will check for any of the allowed special characters using the variable spclChar and then add 10 points onto the finalScore and special character score named, spclScore. After this is completed the code runs through

checking for uppercase, lowercase and numeric characters. For every numeric character 10 points is added to finalScore and for every instance of an upper- or lower-case character 5 points is added to finalScore as well as upperScore and lowerScore respectively.
Finally, once the score has been totalled for all characters in the usrPas and any deductions or additions having been made the user will receive a score rating which will be either Very low, Low, Medium, High, or Very high. The code will rate this between > 20 for Very Low, >20 <=40 for Low, >40 <=70 for Medium, >70 <=80 for High and any other score rating will be Very High. In addition to this the user will receive a detailed report of their score which includes the ratings for the numbers, uppercase, lowercase and special characters they have entered into the password.
The choices I have made about coding conventions
When creating the variables names I have used the coding convention, camel bump where I have name variables that require two words with the first word lower case and the second words first letter in upper case for example, camelBump.
Furthermore, I have used the python convention pe8 in order to organise and properly indent my code so that it's both highly readable and functional, this ensure that my solution is easy to read and works as intended.
Additionally, I have made use of commenting within my code to provide a further detailed analysis of the code and this allows any other developers to read and understand my code in greater depth as well as providing an explanation for aspects of the code that may be misunderstood or hard to read. Having said this, I have made my commenting very basic and so someone with almost layman programming knowledge could understand most if not all of my code comments. This is useful as it allows the user to understand sections of the code and to be able to see why or how their passwords score is being assigned.

The changes I have made during the development process

During the start of my code solution I had to make a choice on how I wanted to validate the users input. I had initially planned to chain if statements to provide validation for their input. However, I decided to use a while loop which saves time and makes my code more efficient and flexible, this has also made my code more readable.
One big change that I have tried to make is that by putting my code into a word document and then exporting it as a pdf has caused some massive indentation issues. These issues have made the code non-functional if copied and ran straight from the pdf. Having put the code in a txt file

however has fixed some of those indentation issues and the code should be functional through there."

Lead examiner commentary:

The learner has demonstrated a mostly accurate and detailed understanding of technical concepts. Valid and mostly supported justification of coding conventions used, and the learner has made logical links between aspects of the solution and the requirements of the scenario.

Valid and mostly supported judgements of the quality and performance of the program. Accurate technical vocabulary used to support arguments.

Mark **band 3** (**9 marks**).

Example 2:

I have made the program according to the scenario requirements. First I had created a flowchart and pseudocode in for me to prepare to make the python program. Flowchart had first allowed to me to break all the process into steps which has helped separate the code without overcomplicating it. I have used shapes such decision, inputs/outputs and process similar to pseudocode which gave me an Idea how to create the English code in pseudocode. After that I created a pseudocode that allowed made me to format all those shapes from flowchart into codes and statements such as decision was IF statements. I have used English programme code so it is easier to understand before full python code was created. However, in the python I will need to use validation methods by using codes to identify the lower or upper case, numeric and special character within the password input

- According to the scenario conditions, the inputted password had a length between 8 and 15. the range, I used an IF statement in the python and put the Input with <=8 and>15 to ensure that the length is between 8 and 15, so it will printed out that the password is correct and within the range. Otherwise, it will print out ERROR and try input the password again until it meets the condition. Although I didn't use a while loop. Along with the IF statement, it made a type of loop ensuring the password has the correct length. To identify the length of password, I had use Len() with the Input password in the IF statement so it will able to identify the exact length and then check it will with the range of <=8 and >=15
- Before adding the points to the total score, I had to first find how many letters or characters are lower or upper case, numerical and special. So I made variables to keep the exact amount inside of them :
  Uppercase = 0
  Lowercase = 0
  Numerical = 0
  Special = 0

  So I used a FOR loop to then use range in order to count up the total amounts for each variable. Then I used an IF statement along with the '.islower' that identify the lowercase letters within the password. Below that, I made a variable with a calculation to keep adding 1 to the total amount of the variable 'lowercase' each time it find lowercase letter. So It count and add 1 to the lowercase variable as each time it count one lowercase letters after the other it will add 1 every time
- In my calculation of adding and deducting the points, I made the variable total to add points by multiplying the points given per lowercase letter ad then add to the total as the Total was set as 0 in the beginning. I did the same for the others afterward. Instead of using multiple variables then add to the total, I just used the same variable total to be stacked by each point after another. So I thought it will make it easier and not over complicate

  The quality of code is code is good as I made used if the IF statements and len code check to ensure the code is according to the scenario. The performance of program when run was performed without mistakes and errors

  # The word document crashed and frozen so I need to screenshot parts of it while it is frozen

If the user were to input something that is a mistake, the program would either register that as error, as a result, it will make you input the password again or shut down the program

Near the finalisation of the program code, I realised there were a few problems where an input password didn't had a numerical digit but it still printed out that points were added in the print statement. Therefore, I added if statement stating that if the numeric numbers were 0 or below, it print out that no points are gained. Otherwise, there was an ELSE statement which below it has a variable with a calculation to add the points which then print out that points were added. In summary, I made the code indicate if the numeric had 0 digits then it wouldn't add points otherwise it will add the points, this helped me avoid a complication in the code.

My program would be better if not for the lack of time. I took too much time of planning the flowchart as it require lots of action in order to place the shapes and arrows. The python is very different compared to pseudocode as it need to have correct code and indexation otherwise the code wold not run at all/ unlike in the pseudocode, the English programme code didn't need to be completely accurate

Lead examiner commentary:

The learner has demonstrated superficial understanding of relevant technical concepts. There is unsupported justification of changes made during the development process and limited justification of coding conventions supported.

Limited judgements about the quality and performance of the program keeps this evaluation in **mark band 1 (3 marks)**.

Summary

Based on performance in this examination series, learners are offered the following advice:

- Apply their knowledge to as many different scenarios as possible. The exam paper will always contain 5 activities which always be the same just the scenario would be different and therefore this will prepare learners to be able to provide answers to the given context under exam conditions.
- Use standard naming conventions throughout the design process and clearly demonstrate this in the flowchart and pseudocode.
- Pseudocode needs to be a detailed yet readable description of what a computer program must do, expressed in a natural language rather than in a programming language if top marks are to be achieved.
- Develop a better understanding of the testing process. Test plan must include normal, abnormal and extreme data. Testing must address errors encountered and how these were overcome. The testing must be iterative, document tests when code is being developed as this will give a true reflection of the development. If required by the scenario any final calculations and outputs from them should be tested and checked.
- Ensure the Program uses accurate validation and error checking procedures throughout, resulting in a robust program that minimises errors and handles unexpected events. This will enhance the completed solution and allow the higher mark bands to be accessed. Programs must address most requirements to gain higher marks.
- The evaluation needs to include a fully supported justification of changes made during the development process, as well as a fully supported justification of coding conventions selected if higher mark bands are to be accessed.