

Task 2: software design and development (part A)

Logan is a technician who has to generate usernames for a school's Wi-Fi service.

Logan wants to write a program that will automatically generate unique usernames for students. The usernames have to be six characters long. The program should generate and display a list of student usernames.

Program analysis

The program will ask how many usernames are to be generated. For each username, the first three letters of the student's first name will be entered and then combined with a random ending from the list below.

The program stores five endings:

ing
end
axe
gex
goh

For a student with the first name David the technician would enter Dav. The program will generate the username by joining Dav to one of the endings listed above. For example the username generated could be Daving.

2a Complete the table by filling in the missing input, process and output.

(3 Marks)

Input	
1.	Enter the number of usernames
2.	Enter the first 3 letters of the student name
Process	
1.	Check length of partial student name
2.	Generate a random number
3.	Add the partial student name with the randomly generated ending from the stored list
Output	
1.	Display the list of generated usernames

- ◆ Check your answers carefully, as you cannot return to part A after you hand it in.
- ◆ When you are ready, hand part A to your teacher or lecturer and collect part B.

Candidate name _____ Candidate number _____

Task 2: software design and development (part B)

Program design

Main Steps: Pseudocode

1. Store the endings
2. Enter the number of students
3. Start fixed loop for each student
 4. Enter first three letters of student's name
 5. Generate random number
 6. Generate username
 7. Display the username
8. End Loop

REFINEMENTS

- 4.1 Start conditional loop
 - 4.2 Get the first three letters of student's name
 - 4.3 If the length of the name is not equal to 3 then
 - 4.4 Display an error message
 - 4.5 End If
 - 4.6 Repeat until the name entered is 3 characters long
-
- 6.1 If the first random number was generated add the first stored ending to the end of the first three letters of the student's name
 - 6.2 If the second random number was generated add the second stored ending to the end of the first three letters of the student's name
 - 6.3 If the third random number was generated add the third stored ending to the end of the first three letters of the student's name
 - 6.4 If the fourth random number was generated add the fourth stored ending to the end of the first three letters of the student's name
 - 6.5 If the fifth random number was generated add the fifth stored ending to the end of the first three letters of the student's name

- 2b Using the program design and refinements, implement the program in a language of your choice. Ensure the program matches the pseudocode provided.

(15 marks)

Print evidence of your program code.

```
1 #2019 Example Solution
2
3 endings=[""]*5
4 noOfStudents =0
5 partName =""
6 randomNo =0
7 userName = ""
8
9 import random
10
11 #Store Endings
12 endings[0]="ing"
13 endings[1]="end"
14 endings[2]="axe"
15 endings[3]="gex"
16 endings[4]="goh"
17
18 #Enter the number of students
19 noOfStudents=int(input("Enter the number of students"))
20
21 #Start fixed loop for each student
22 for counter in range(noOfStudents):
23
24     #Enter and validate first 3 letters of name
25     partName = input("Please enter first 3 letters of student name")
26     while len(partName)!=3:
27         partName = input("Re-enter only the first 3 letters")
28
29     #Generate random number between 0 and 4 to pick random ending
30     randomNo = random.randint(0,4)
31
32     #Generate username - concatenate name to random ending
33     userName = partName + endings[randomNo]
34
35     #Display each username
36     print("Student " + str(counter + 1)+ ":" + userName)
37
38 #End loop
39
```


2c Your program should be tested to ensure it will only accept 3 characters.


Complete the test table below

(2 marks)

Type of test	User input	Expected result	Actual result
Normal	Ala	Input accepted	Printout of final output to show that input is accepted.
Exceptional	Harry	Error message displayed	Printout to show that an error message is generated.

Evidence:

```
Powered by  trinket
Enter the number of students 1
Please enter first 3 letters of student name Ala
Student 1:Alaaxe
```

```
Powered by  trinket
Enter the number of students 1
Please enter first 3 letters of student name Harry
Re-enter only the first 3 letters █
```


2d Test your program using the following student names.

Chris
Christina
Christopher
Chrethe
Chrisoula
Christie

Provide evidence of the inputs and outputs to show that you have completed the test.

(1 mark)

All invalid

```
Powered by  trinket
Enter the number of students 6
Please enter first 3 letters of student name Chris
Re-enter only the first 3 letters Christina
Re-enter only the first 3 letters Christopher
Re-enter only the first 3 letters Crethe
Re-enter only the first 3 letters Chrisoula
Re-enter only the first 3 letters Christie
Re-enter only the first 3 letters Chr
Student 1:Chrgex
Please enter first 3 letters of student name █
```

Candidate name _____ Candidate number _____

2e With reference to your code and testing, evaluate your own 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:

- Asks the user for the number of unique usernames to generate
- It stores the endings in an array
- Generates a random number to pick a random ending to add onto the first 3 letters of the users name.

Efficiency of your code (1 mark)

My code is efficient as I have used an array to store the endings instead of single variables

I have used a fixed loop to reduce the number of lines of code

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.

I tested using names that were over 3 letters long and the program asked the user to re-enter.

I also checked that the endings were being randomly selected

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 _____