# Project Assessment: Code an application

## Criteria

### Unit code and name

ICTPRG302 – Apply introductory programming techniques (1)

### Qualification/Course code and name

Teaching staff/student to select the correct qualification the student is enrolled in from the below dropdown list:

ICT40120 | Certificate IV in Information Technology (3)

## Student details

### Student number

881275421

### Student name

Ming Chi Hsiao

## Assessment declaration

*Note: If you are an online student, you will be required to complete this declaration on the TAFE NSW online learning platform when you upload your assessment.*

This assessment is my original work and has not been:

* plagiarised or copied from any source without providing due acknowledgement.
* written for me by any other person except where such collaboration has been authorised by the Teacher/Assessor concerned.

### Student signature and date

Ming Chi Hsiao 01/04/2024

Version: 20211028

Date created: 10 August 2021

Date modified: 28 October 2021

For queries, please contact:

Technology and Business Services SkillsPoint

Ultimo

© 2021 TAFE NSW  
RTO Provider Number 90003 | CRICOS Provider Code: 00591E

This assessment can be found in the: [Learning Bank](https://share.tafensw.edu.au/share/access/searching.do?doc=%3Cxml%2F%3E&in=P7ac4831b-430a-4b8d-8b56-f7b32ed5b9cf&q=&type=standard&sort=rank&dr=AFTER)

The contents in this document is copyright © TAFE NSW 2021 and should not be reproduced without the permission of TAFE NSW. Information contained in this document is correct at the time of printing: 31 May 2024. For current information please refer to our website or your Teacher/Assessor as appropriate.

## Assessment instructions

Table 1 Assessment instructions

| Assessment details | Instructions |
| --- | --- |
| **Assessment overview** | The objective of this assessment is to assess your knowledge and performance in creating simple programs using introductory programming techniques. |
| **Assessment event number** | 2 of 2 |
| **Instructions for this assessment** | This is a project-based assessment that assesses your knowledge and performance of the unit.  This is a project-based assessment that assesses the student on their knowledge and performance of the unit.  This assessment is in five parts:   1. The project brief 2. Design the application 3. Develop the program 4. Testing 5. Program debugging   And is supported by:   * Assessment Checklist * Assessment feedback (not included here) * [ICTPRG302\_AE\_Pro\_Appx](https://share.tafensw.edu.au/share/items/87a36bb7-43ba-4be5-a2fb-526c7cfaf328/0/?attachment.uuid=713f8c6e-ea33-4ac6-94a7-b44c90bdd431) which contains:   + Gelos Enterprises Software Design Form   + accounts.txt (test data)   **Note**: This assessment may contain links to external resources. If a link does not work, copy and paste the URL directly into your browser. |
| **Submission instructions** | On completion of this assessment, you are required to submit it to your Teacher/Assessor for marking. Where possible, submission and upload of all required assessment files should be via the TAFE NSW online learning platform.  It is important that you keep a copy of all electronic and hardcopy assessments submitted to TAFE and complete the assessment declaration when submitting the assessment. |
| **What do I need to do to achieve a satisfactory result?** | To achieve a satisfactory result for this assessment you must answer all the questions correctly.  If a resit is required to achieve a satisfactory result it will be conducted at an agreed time after a suitable revision period. |
| **What do I need to provide?** | * TAFE NSW student account username and password. If you do not know your username and password, contact your campus or service centre on 131601. * Computer or other device with word processing software, internet access and a camera and microphone or mobile phone to make recordings. This course will require you to record yourself for certain assessment tasks. These recordings are for assessment purposes only and will only be shared with your assessing teacher. * Programming software – either Python, Unreal Engine or other programming software used in class studies. * Reference documents. |
| **What the Teacher/Assessor will provide** | Access to this assessment and learning resources, including the student workbook and any supporting documents or links.  Equipment/resources, including:   * If working in a classroom, computer with camera and microphone. * Software – Python, Unreal Engine (C++) or other programming software used in the learning activities. * [ICTPRG302\_AE\_Pro\_Appx](https://share.tafensw.edu.au/share/items/87a36bb7-43ba-4be5-a2fb-526c7cfaf328/0/?attachment.uuid=713f8c6e-ea33-4ac6-94a7-b44c90bdd431) which contains:   + Gelos Enterprises Software Design Form   + accounts.txt (test data) |
| **Due date**  **Time allowed**  **Location** | Refer to UAG  Five hours (indicative only)  Assessment may be completed in a class or online environment. |
| **Assessment feedback, review or appeals** | In accordance with the TAFE NSW policy *Manage Assessment Appeals,* all students have the right to appeal an assessment decision in relation to how the assessment was conducted and the outcome of the assessment. Appeals must be lodged within **14 working days** of the formal notification of the result of the assessment.  If you would like to request a review of your results or if you have any concerns about your results, contact your Teacher/Assessor or Head Teacher. If they are unavailable, contact the Student Administration Officer.  Contact your Head Teacher/Assessor for the assessment appeals procedures at your college/campus. |

## Specific task instructions

The instructions and the criteria in the tasks and activities below will be used by the Teacher/Assessor to determine if the student has satisfactorily completed this assessment event. Use these instructions as a guide to ensure the student demonstrates the required knowledge and skills.

To complete this part of the assessment, you are required to evidence your participation in an interactive role-play. This will be achieved by viewing an interactive video and then capturing your responses in a recording.

Refer to the [Observation Checklist](bookmark://ObsChklst) to understand what skills you need to demonstrate in this section of the assessment. This checklist outlines the assessment criteria your Teacher/Assessor will be marking you on.

Once completed, the recorded evidence will be submitted via the online platform to the Teacher/Assessor for marking.

This digital recording may be either an audio file (sound only) **or** video file (video and audio). You may use your computer webcam and capture software or your mobile phone. Ensure you have access to the required equipment and resources.

If space or bandwidth is limited, create an audio file rather than video. Video file uploads are limited to 1Gb.

TIP: The following may be helpful: [video recording instructions (pdf)](https://share.tafensw.edu.au/share/file/744af7d4-a241-45e2-adb0-0e13f2fe4950/1/VideoAssessmentInstructions.pdf). This one-page includes useful tips, links to resources, and a demonstration video.

Refer to the scenario outline and start with task 1.1 to complete this assessment part.

**The scenario**

You have been employed as an ICT trainee with [Gelos Enterprises](https://share.tafensw.edu.au/share/items/d0b458dc-3922-409d-b1fe-9a2f785f4a38/0/GelosEnterprises.zip/index.html). To further your training, the company has asked Christina Kaiser, the Software Development Team Leader, to train you in all aspects of programming and you have been assigned to a new programming project.

Gelos requires a simple login program with a menu of options:

* **Login** – for users who have previously registered. Username and password to be checked for validity.
* **Register** – create a new user account.
* **Passwords** – new users given the option to enter their own password or generate one.
* **Generated passwords** – user given the option to choose the password character types – numbers, symbols or letters. The default password length should be applied, but users should also be allowed to choose how many characters
* **Save file** – usernames and passwords should be saved to a text file **accounts.txt**
* **Exit** – delay for 2 seconds before exit
* **View accounts** – to display user account information from the accounts.txt file (assuming that only admin have access to this program).

You will need to interview your supervisor, Christina Kaiser to get all of the information you need and clarify the details of the project.

For the purpose of this assessment, an interactive conversation featuring Christina Kaiser has been pre-recorded. Refer to Task 1.1 – **interactive video roleplay**.

To complete this task, you will need to access and view this [**interactive video**](https://share.tafensw.edu.au/share/items/87a36bb7-43ba-4be5-a2fb-526c7cfaf328/0/?attachment.uuid=71b797da-dd1e-4658-ac27-27aea3da8ac9).

## Part 1: The project brief

**Task 1.1: Confirm work brief and clarify requirements – interactive video roleplay**

**Getting ready for the roleplay:**

* Look at the partially completed software design document, so you know what details are missing from the project brief.
* You will need to ask Christina three questions regarding the project. Be clear and specific in your request for each piece of information and use correct terminology – it is not enough to say, 'what about passwords?' and assume they will know what you mean.
* See the table below for a list of what these questions should be about and in what order you should ask them. As this is a pre-recorded video, you will need to ask the questions in the same order as they appear in the table below.

Use this table to prepare and write down your questions to remind yourself of what to ask:

Table 2 Questions template

|  |  |  |
| --- | --- | --- |
| Ask a question about: | Question to ask Christina | |
| The programming language to be used for the project | What is the programming language to use for this project? And what application we will use this project for? |
| Confirm the format of the passwords to be generated | Any requirement for the password formation (all numbers, all chars, or mix) What is the requirement for the password to be generated. Can they choose it? |
| Confirm the length of the password | The length of the password, what is the default length of password we should generate, and the minimum length and maximum length of password we should allow the customer to create. |

## Create your recordings

To complete this task, you will need to access and view this [**interactive video**](https://share.tafensw.edu.au/share/items/87a36bb7-43ba-4be5-a2fb-526c7cfaf328/0/?attachment.uuid=71b797da-dd1e-4658-ac27-27aea3da8ac9).

The interactive video will have three pause points where you will ask your questions to Christina Kaiser. You may submit your three questions **within one recording** or **separate files**. If in one recording, leave a 10 second gap between your questions so the assessor can clearly identify your three different questions.

**Recording process:**

* Activate the [**interactive video**](https://share.tafensw.edu.au/share/items/87a36bb7-43ba-4be5-a2fb-526c7cfaf328/0/?attachment.uuid=71b797da-dd1e-4658-ac27-27aea3da8ac9)**.**
* The video will play, Christina will speak, and a message will appear asking you to record your question. Press pause on the video.
* Use your device to start recording and proceed to record your first question.
* When finished recording, press pause on your recording device.
* Return to the video and press play to continue.
* Repeat this process until you have recorded all three questions.

At the end of the interactive roleplay, you have the opportunity to play back your recordings. If you are not happy with your recordings, you can restart the interactive video and re-record your questions.

If you are happy with your recordings, save the file(s), upload in the space provided and click ‘Submit’.

**Task 1.2:** Complete **the Software Design Document**

Once you have met with Christina and confirmed the requirements of the project, complete the required sections of the Gelos Software Design Document.

Save the program specification form as per company naming conventions (**Yourname\_Software\_Design\_Document.docx**).

## Part 2: Design the application

**Task 2.1: Coding rules**

Before commencing on your design, it’s a good idea to refresh your memory on the rules of coding. In the space provided in the Gelos Software Design Document (*Coding Rules*), list five basic rules for the programming language you are using to complete this project. Include a brief explanation of each rule (approximately 20-30 words each).

**Task 2.2: Design data library variables**

You will need to use variables in your application. In the space provided in the Gelos Software Design Document (*Data Library Variables*), list all of the variables needed for your design and identify how these variables will be used.

**Task 2.3: Identify variable scope**

For each of the operators and expressions to be used in your program, nominate the scope for that variable. Record these details in the space provided in the Gelos Software Design Document (*Variable Scope*). For example: global, username.

**Task 2.4: Inspect test data**

To enable testing of the program once it has been created, a data file containing test data has been provided. Inspect the file **accounts.txt** to familiarise yourself with this data. These details have already been recorded in the Gelos Software Design Document (*Program Test Data*).

**Task 2.5: Develop an algorithm**

Develop an algorithm using a suitable tool such as pseudocode or a flowchart.

Include the following:

1. Iterate through the accounts.txt file to read each record. Use a logical operator in the iteration, either when entering or exiting the loop.
2. Use a logical operator to allow the users to select a menu option using either upper-case or lower-case characters (for example: a or A).
3. Include other steps in a logical sequence.

Paste a copy of the pseudocode or flowchart in the space provided in the Gelos Software Design Document (*Program Algorithm*).

## Part 3: Develop the program

**Task 3.1: Develop the program**

Code the algorithm in the programming language you are learning in this unit, using the correct syntax, including:

1. A comment at the very top of the program that includes your name, date and purpose of program.
2. Use appropriate function libraries.
3. Comment your code throughout.
4. For saving (write) and viewing (read) account details using file (accounts.txt), make use of control structures (selection and iteration as required).
5. Use expressions, constructs and logical operators to display main menu, log in, generate random password and create accounts. Remember to allow the users to select a menu option using either upper-case or lower-case characters (for example: a or A).
6. Include data structures in the creation of the login process.
7. Apply string manipulation by concatenating the elements of the randomly generated password.

Paste a copy of the program code (or blueprints) into the Gelos Software Design Document (Program Code).

**Task 3.2: Present to supervisor, obtain feedback and sign off**

Now that you have designed and created the program, you will need to present it to your supervisor for feedback and approval. Submit these assessment pages and your Gelos Software Design Document via the online learning platform and await feedback before proceeding with the testing.

Table 3: Feedback

|  |
| --- |
| Feedback from supervisor (assessor) |
| *To be completed by the assessor* |

**Task 3.3: Modify your program**

Once you receive the feedback, make any changes or modifications that were recommended and update your Gelos Program Design Document (*Changes Required from Feedback*) to record the changes made.

**Note to student** – if you have some spare time while awaiting feedback, you can complete the small programming task in Part 5 of this assessment.

## Part 4: Testing

**Task 4.1: Test and record results**

Test and record the results of your testing in the Software Test Report section of the Gelos Software Design Document. Be sure to include tests for all four menu options, as well as for menu options that do not exist, for example, try entering Q and test the results.

For each test, take a screenshot and save the image using a suitable file name. Record the filename in the Screenshot Filename column of the Software Test Report table.

When all testing is complete, create a zip file containing all of your screenshot images. Name this file **YourName\_Testing.jpg** for inclusion in your final submission.

**Task 4.2: Software evaluation**

Now that you have completed your program and testing, you will need to evaluate your solution and ensure it meets the specifications requested by the client. In the Software Evaluation section of the Gelos Software Design Document, complete the table with the specification and the evaluation of your solution.

**Task 4.3: Programmers checklist**

Complete the programmer’s checklist in the Gelos Software Design Document to ensure you have completed all tasks.

## Part 5: Program debugging

After completing the program for the login, Christina Kaiser has asked for your assistance with another small job, fixing a program that was written by another trainee which is not correct.

The trainee was asked to create an application that determines the average mark for a student based on their marks from five different subjects (see code below).

The instructions stated that the application must do the following:

* Ask the user to input the marks for the five subjects in a list/array.
* The program must ensure that the marks are between 0 and 100
* Display the list/array of marks entered.
* Find the sum of all the marks in the list (all five subjects) and display the output as:
  + The sum of your marks is: [sum]
* Find the average of all the marks in the list (all five subjects) and display the output as:
  + The average of your marks is: [average mark]

Table 4: Code for marks program

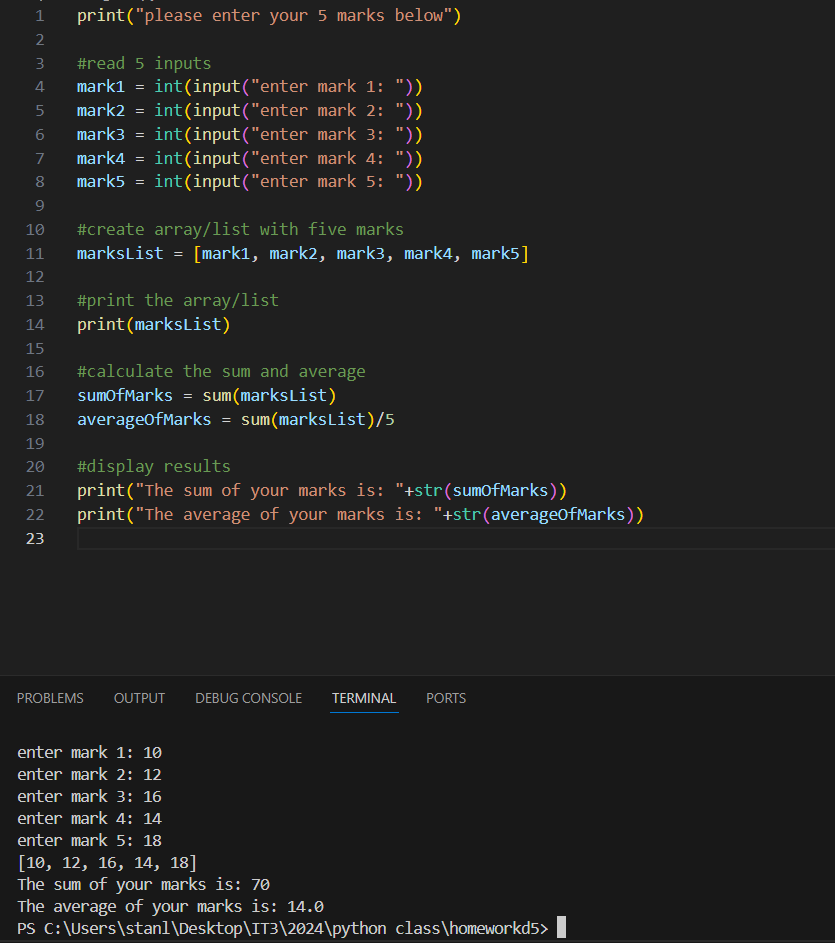
| Code for marks program |
| --- |
| print("please enter your 5 marks below")  #read 5 inputs  mark1 = int(input("enter mark 1: "))  mark2 = int(input("enter mark 2: "))  mark3 = int(input("enter mark 3: "))  mark4 = int(input("enter mark 4: "))  mark5 = int(input("enter mark 5: "))  #create array/list with five marks  marksList = [mark1, mark2, mark3, mark4, mark5]  #print the array/list  print(marksList)  #calculate the sum and average  sumOfMarks = sum(marksList)  averageOfMarks = sum(marksList)/5  #display results  print("The sum of your marks is: "+str(sumOfMarks))  print("The average of your marks is: "+str(averageOfMarks)) |

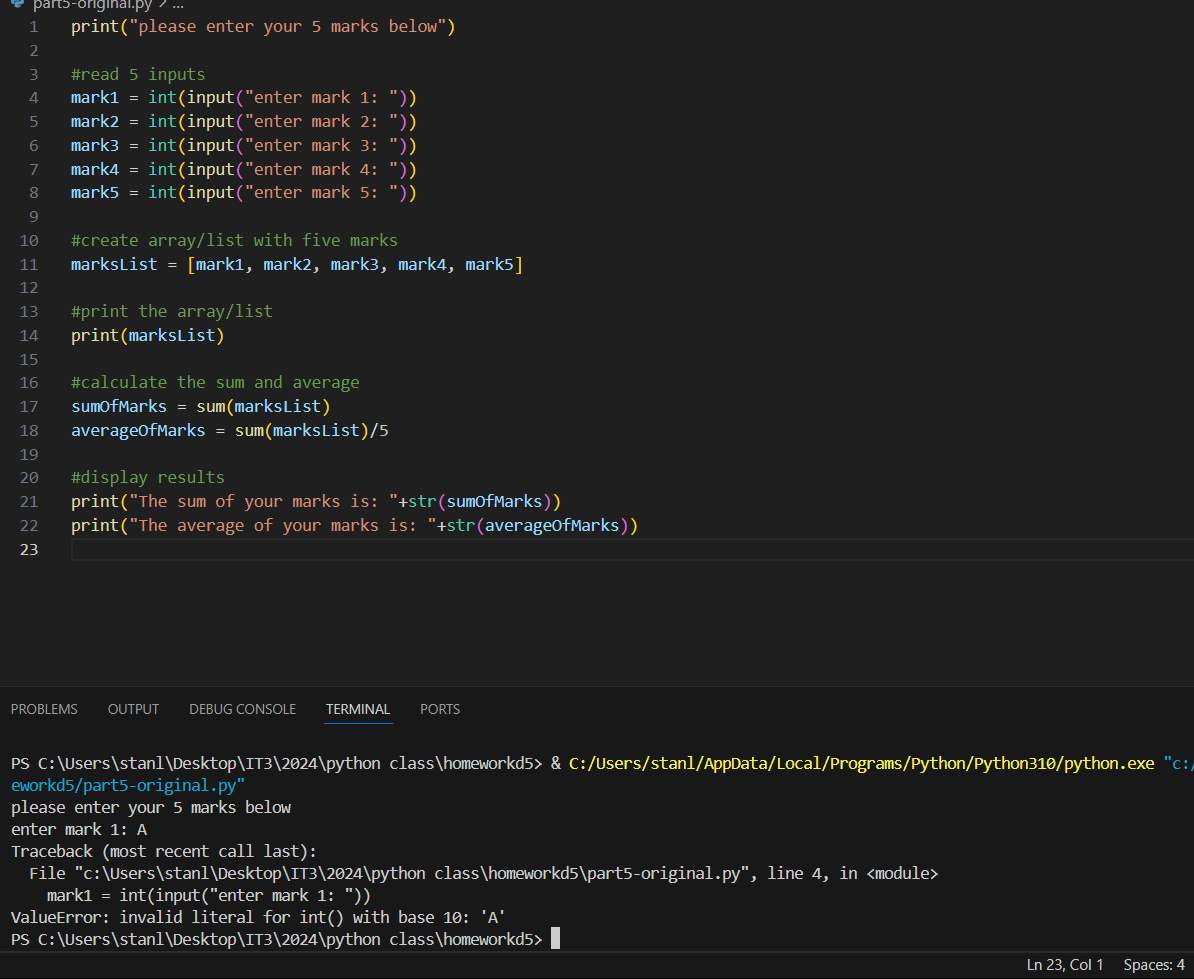
#### Test data

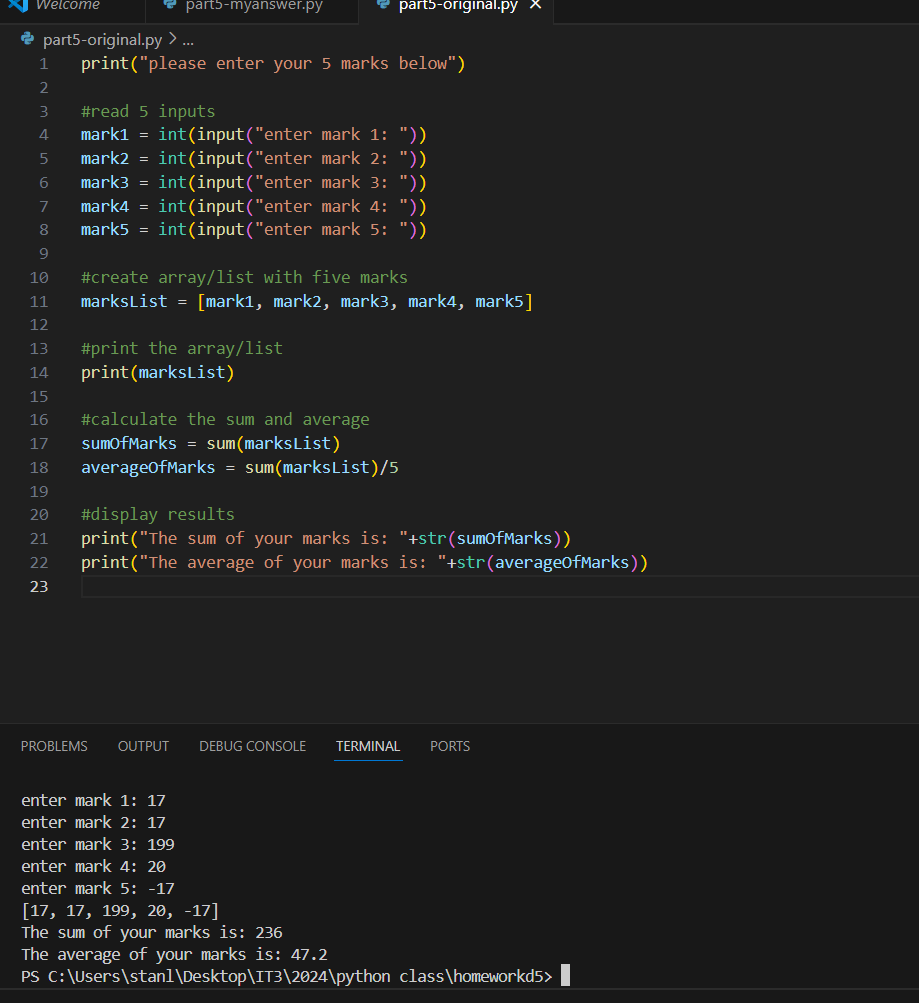
Table 5: Test data

| mark1 | mark2 | mark3 | mark4 | mark5 | Expected sum | Expected average | Comments |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 10 | 12 | 16 | 14 | 18 | 70 | 14 | The result is correct |
| A | B | D | A | C | n/a | n/a | Exception error on input |
| 17 | 17 | 199 | 20 | -17 | 90 | 18 | The sum is 236, the average is 47.2, but this program should not allow for -17 and 199 to be the mark. |

1. Create another test case using the samples above as a guide.







1. Use debugging and problem-solving techniques to detect and correct errors in the code (use the test cases to confirm the design specifications). This must include examining the contents of the variables. Document the changes you made by including comments in the code, using the correct syntax.

Table 6: Corrected code for marks program

| Corrected Marks code |
| --- |
| #Create a function to make user only be able to input integer between 0-100, otherwise error message will be provide.  def mark(i):     while True:        try:          print("please enter mark"+str(i)+" below")          mark = int(input("type number between 0-100: "))            if 0>mark or mark>100:              raise ValueError          else:              return mark      except ValueError:          print("Invalid Integer, please enter number between 0-100 ")    print("please enter your 5 marks below")  #call the mark function to read each function  mark1=mark(1)  mark2=mark(2)  mark3=mark(3)  mark4=mark(4)  mark5=mark(5)    #create array/list with five marks  marksList = [mark1, mark2, mark3, mark4, mark5]  #print the array/list  print(marksList)  #calculate the sum and average  sumOfMarks = sum(marksList)  averageOfMarks = sum(marksList)/5  #display results  print("The sum of your marks is: "+str(sumOfMarks))  print("The average of your marks is: "+str(averageOfMarks)) |

1. For each of the test data that generates an error, include labelled screenshots below of your debugging techniques. (2 screenshots)

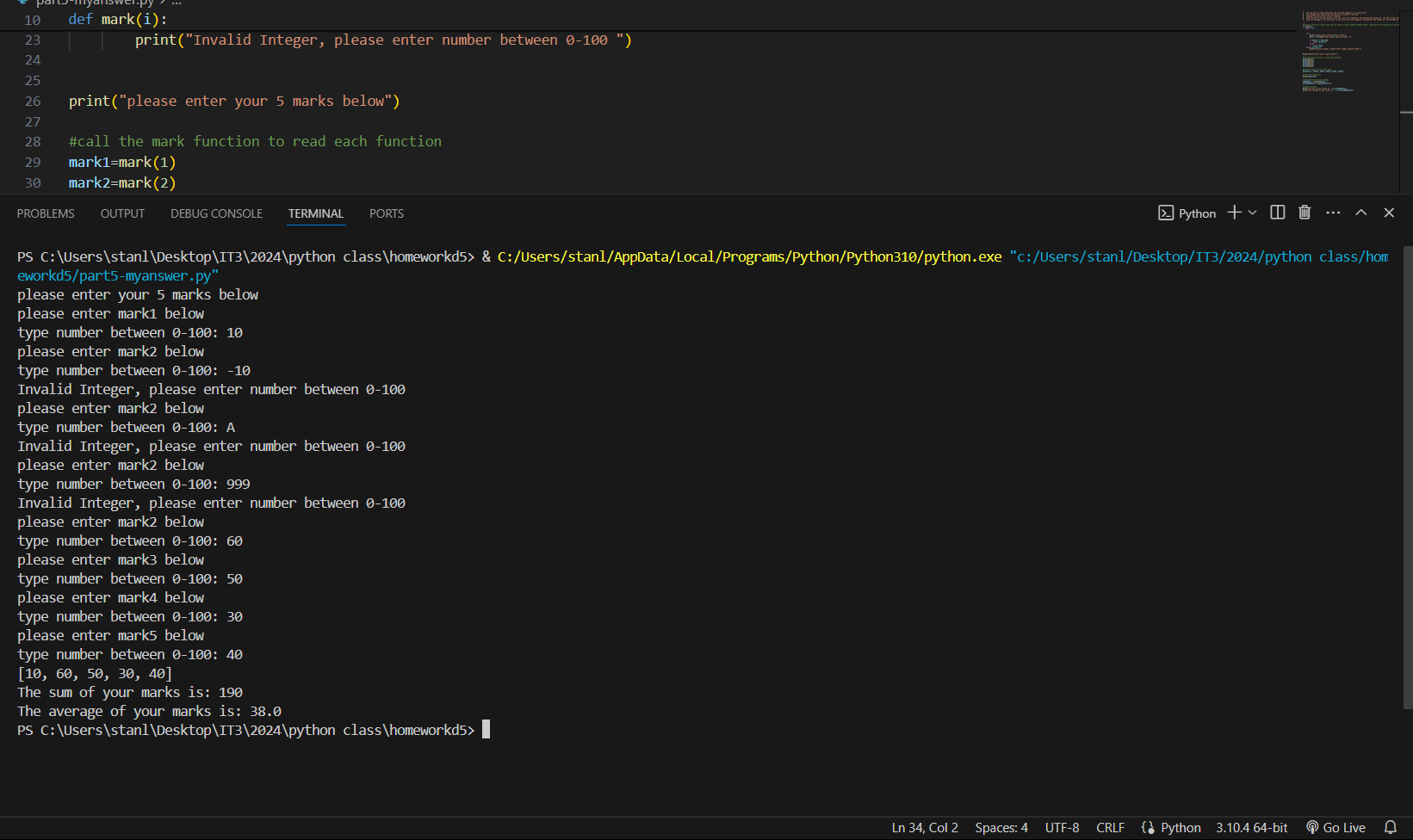
Table 7: Debugging screenshots

| Debugging screenshots |
| --- |
| *Paste screen**shot here* |
| *Paste screen**shot here* |

**Submission**

Now that you are all done, you need to submit your files, including:

* These assessment pages
* Your interview video recordings
* The completed Gelos Software Design Document
* The zip file containing your testing screenshots.



## Observation Checklist

The Observation Checklist will be used by your Teacher/Assessor to mark your performance in the previous event type/s. Use this Checklist to understand what skills you need to demonstrate in the role play scenario, presentation or demonstration. The Checklist lists the assessment criteria used to determine whether you have successfully completed this assessment event. All the criteria must be met. Your demonstration will be used as part of the overall evidence requirements of the unit. The Teacher/Assessor may ask questions while the demonstration is taking place or if appropriate directly after the task/activity has been completed.

Table 8: Observation checklist

| TASK/STEP # | Instructions | S | U/S | Assessor Comments |
| --- | --- | --- | --- | --- |
| 1 | Used effective questioning and listening techniques to confirm requirements, articulate complex concepts using common programming terminology. |  |  | *Date of Observation:* |

**Assessment Checklist**

The student's copy of the Assessment Checklist will be used by you to capture evidence of their performance in this Part 1 of this assessment. This checklist outlines all the required criteria you will be marking the student on. All criteria described in the Assessment Checklist must be met. The following checklist contains benchmark responses for you to use when assessing to ensure the reliability of judgement.

Table 9: Assessment checklist

| TASK/STEP # | Instructions | | S | U/S | Assessor Comments |
| --- | --- | --- | --- | --- | --- |
| Part 2  Task 2.1 | | Student identified programming standards and guidelines according to task requirements |  | |  | Date of Observation:  Assessors are to record their observations in enough detail to demonstrate their judgement of the student's performance against the criteria. |
| Task 2.2 | | Student applied variables |  | |  |  |
| Task 2.3 | | Student has applied variable scope |  | |  |  |
| Task 2.6 | | Student has designed an algorithm in response to basic program specifications |  | |  |  |
|  | | 1. Student created expressions in selection and iteration constructs using logical operators |  | |  |  |
|  | | 1. Student used logical operators for menu selections |  | |  |  |
|  | | 1. Student applied language syntax in sequence, selection and iteration constructs |  | |  |  |
| Part 3  Task 3.1 | | Student created code using language data types, operators and expressions and developed application to meet program specification |  | |  |  |
|  | | 1. Student added comment to top of code with name, date and purpose of program |  | |  |  |
|  | | 1. Student used appropriate library functions |  | |  |  |
|  | | 1. Student clarified meaning of code using commenting techniques |  | |  |  |
|  | | 1. Student used selection and iteration constructs using logical operators and made use of control structures. |  | |  |  |
|  | | 1. Student creates expressions in selection and iteration constructs using logical operators to display menu, log in, generate password and create accounts. |  | |  |  |
|  | | 1. Student creates and uses data structures in the creation of the login process. |  | |  |  |
|  | | 1. Student applied string manipulation by concatenating the elements of the password. |  | |  |  |
| Task 3.2 | | The student presented their program for feedback and approval. |  | |  |  |
| Task 3.3 | | Student modified program as per feedback and obtained sign off |  | |  |  |
| Part 4  Task 4.1 | | The student used debugging techniques to detect and correct errors; created and confirmed simple tests; documented actions taken to fix coding errors. |  | |  |  |
| Task 4.2 | | Student carried out evaluation to confirm application meets initial specifications |  | |  |  |
| Part 5  Task 5.1 | | Student examined variable contents and use debugging techniques to detect and correct errors, documenting changes made in the code |  | |  |  |
| Task 5.2 | | Student documented test data errors (by providing screenshots) |  | |  |  |

## Assessment feedback

*NOTE: This section must have the Teacher/Assessor and student signature to complete the feedback. If you are submitting through the TAFE NSW online learning platform, your Teacher/Assessor will give you feedback via the platform.*

### Assessment outcome

Satisfactory

Unsatisfactory

**Assessor feedback**

Has the assessment declaration for this assessment event been signed and dated by the student?

Are you assured that the evidence presented for assessment is the student’s own work?

Was reasonable adjustment in place for this assessment event?

*If yes, ensure it is detailed on the assessment document.*

*Comments*:

### Assessor name, signature and date

### Student acknowledgement of assessment outcome

*Would you like to make any comments about this assessment?*

### Student name, signature and date