SINGAPORE UNIVERSITY OF SOCIAL SCIENCES

ICT 133
Structured Programming

Tutor-Marked Assignment

January 2024 Presentation

TUTOR-MARKED ASSIGNMENT (TMA)

This assignment is worth 24 % of the final mark for ICT133, Structured Programming.

The cut-off date for this assignment is **Sunday, 17 March 2024, 2355 hours**.

Assignment Requirements:

- Do **NOT** define classes for this TMA.
- **Unless specified in the question**, **you CANNOT use packages not covered in this module, e.g., re, collections, numpy, pandas etc.**
- All functions must be documented. Provide sufficient comments to your code and ensure that your program adheres to good programming practices such as not using global variables.
- **Do not use the exit() function.**
- Failing to do so can incur a penalty of as much as 50% of the mark allotted.

Submission Details:

- Use the template word document provided - **SUSS_PI_No-FullName_TMA.docx.** Rename the file with your SUSS PI and full name join with "_TMA" e.g., "**SUSS_PI-TomTanKinMeng_TMA.docx**" (without the quotes).
- Include the following particulars on the first page of the word document, on separate lines: **Course Code, SUSS PI No., Your Name, Tutorial Group and Submission Date**.
- Copy and paste the source code of each program you write in **text** format. Submit screenshots for **only** output of your program, where applicable.
- If you submit the source code as a screenshot, your code will not be marked. That is, **screenshot code will be awarded zero mark.**
- Submit your solution in the form of a **single MS Word document in Canvas Assignment page. Do NOT submit as a pdf document**. You will be penalised if you fail to submit a word document.
- **The word document must be submitted to your respective T group. Besides this, you are required to upload your source code to Vocareum.**

Question 1 (25 marks)

This question covers materials in Seminar 1 and 2. Use and express selection structure for this question. **Functions, repetition, or collections are not necessary for this question**.

(a) MOF has announced a $1.1 billion Cost-of-Living Support Package on 28 September 2023. This includes a $0.8 billion enhancement to the Assurance Package (AP), to provide more relief for Singaporean households, especially lower- to middle-income families.

Singaporeans Aged 21 Years and Above in 2024				
Payment	Owns 0 to 1 property			Owns more than 1 property
	Assessable Income for Year of Assessment 2022			
	Up to $34,000	More than $34,000, and up to $100,000	More than $100,000	
AP Cash	$600	$350	$200	$200
AP Cash Special Payment	$200	$150	-	-
Total	$800	$500	$200	$200

Table 1

Write an AP Calculator program that helps Singapore citizens find out the estimated amount they may receive from the Government.

Your program will read in the user's birth year, number of properties owned, and assessable income for 2022. Assume user will enter valid input.

Based on Table 1, your program will determine and display the total eligible amount, with details of AP Cash and AP Cash Special Payment (if qualify). Study the sample executions below carefully:

```
Sample 1:
Enter year of birth: 1980
Do you own more than 1 property? (Y/N): N
Assessable Income (2022): 80000
You will receive total $500 ($350 AP Cash + $150 AP Cash Special Payment)

Sample 2:
Enter year of birth: 1980
Do you own more than 1 property? (Y/N): Y
You will receive total $200 ($200 AP Cash)

Sample 3:
Enter year of birth: 2004
You are not eligible for AP package
```

(15 marks)

(b) Write a program that reads in a time in 24-hour clock format (e.g. 2359). The program displays a greeting with the time in 12-hour format. The greeting according to the time is as follows:

Good morning from 0000 to 1159
Good afternoon from 1200 to 1759
Good evening from 1800 to 2359

Assume user will enter valid input. Study the sample executions below carefully:

```
Sample 1
Enter time in 24-hour format: 1005
Good morning! Time is 10.05am

Sample 2
Enter time in 24-hour format: 1210
Good afternoon! Time is 12.05pm

Sample 3
Enter time in 24-hour format: 1805
Good evening! Time is 6.05pm

Sample 4
Enter time in 24-hour format: 0001
Good morning! Time is 12.01am
```

(10 marks)

For each part, submit and paste screenshots of at least three program executions, with different input.

Question 2 (25 marks)

This question covers materials up to Seminar 3. Make use of functions, selection, and repetition structures. **NO data structures like set, list, tuple, or dictionary should be used for this question.** Keep the program modular by defining other functions if necessary.

This question is divided into 3 parts. Submit separate python codes for each part. Paste screenshot of a program executions that covers all scenarios.

(a) Write a Python program that allows 2 players to play the following dice game.

i. At start of the game, prompt for the following:
   - Two unique names representing the 2 players
   - Number of rounds to play
ii. In each round, player will roll the dice
iii. Player with bigger dice value wins this round
iv. Display the players' dice values and current score
v. Repeat step ii to iv, until all rounds are played
vi. If there is no winner, conclude the game as a draw

```
Sample 1
Enter player 1 name: Jack
Enter player 2 name: Jim
No of rounds to play: 3

Round 1 - Jack 3 : Jim 4
Current Score – Jack 0 : Jim 1

Round 2 - Jack 2 : Jim 1
Current Score – Jack 1 : Jim 1

Round 3 - Jack 4 : Jim 5
Current Score – Jack 1 : Jim 2

Jim is the winner!!
```

```
Sample 2
Enter player 1 name: Jack
Enter player 2 name: Jim
No of rounds to play: 3

Round 1 - Jack 3 : Jim 4
Current Score – Jack 0 : Jim 1

Round 2 - Jack 2 : Jim 1
Current Score – Jack 1 : Jim 1

Round 3 - Jack 5 : Jim 5
Current Score – Jack 1 : Jim 1

It's a draw!!
```

(10 marks)

(b) Enhance Q2(a) to allow 2 players to play a modified dice game:

i. At start of the game, prompt for
   - Two unique names representing the 2 players
   - Number of rounds to play

ii. In each round, player will roll the dice

iii. Player with bigger dice value wins this round

iv. Display the players' dice values and number of consecutive wins

v. The game ends when a player wins 2 consecutive rounds

vi. Repeat step ii to v, until all rounds are played

vii. If there is no winner, conclude the game as a draw

```
Sample 1
Enter player 1 name: Jack
Enter player 2 name: Jim
No of rounds to play: 10

Round 1 - Jack 3 : Jim 4
Consecutive wins – Jack 0 : Jim 1

Round 2 - Jack 2 : Jim 1
Consecutive wins – Jack 1 : Jim 0
```

```
Round 3 - Jack 4 : Jim 4
Consecutive wins – Jack 0 : Jim 0

Round 4 - Jack 6 : Jim 5
Consecutive wins – Jack 1 : Jim 0

Round 5 - Jack 4 : Jim 1
Consecutive wins – Jack 2 : Jim 0

Jack is the winner!!
```

Sample 2

```
Enter player 1 name: Jack
Enter player 2 name: Jim
No of rounds to play: 3

Round 1 - Jack 3 : Jim 4
Consecutive wins – Jack 0 : Jim 1

Round 2 - Jack 2 : Jim 1
Consecutive wins – Jack 1 : Jim 0

Round 3 - Jack 5 : Jim 5
Consecutive wins – Jack 0 : Jim 0

It's a draw!!
```

(10 marks)

(c) Modify the program in Q2(b) to perform the following:

- After each game, prompt whether players wish to play again with this prompt: "Play again? (y/n)". If yes, repeat the game from step ii - vii.

- Track the winner for each game so that a summary can be printed when players decided to stop. E.g. "Overall score: Jack 3 : Jim 4"

(5 marks)

Question 3 (25 marks)

This question covers materials up to seminar 4. **The data structure to use is list or nested list. No dictionary collection is required for this question.** Keep the program modular by defining necessary functions.

Read both part (a) and (b) before attempting the question. You can submit Q3(a) and (b) together.

(a) Write a function *countRepeatingChar(word: str) -> int:* that returns the highest number of the "repeating letter" in the given parameter word. Assume the given parameter word is in lowercase.

For example, the function returns these values with the following parameters:

- countRepeatingChar('assistants') → 4
- countRepeatingChar('that') → 2
- countRepeatingChar('business') → 3
- countRepeatingChar('count') → 1

(10 marks)

(b) Write a function *digest(words: str) -> list:* that reads the words from the given parameter and returns a nested list of words with the same number of repeating letters. A sample main( ) program on how to call the *digest( )* function and the expected nested list is shown below:

```
def main():
    words = 'assistants,repeated,that,function,count,letters,suss,business,word'
    print( digest(words) )
    """
      output from print( digest(words) )

      [ ['count', 'word'],
        ['that', 'function', 'letters'],
        ['repeated', 'suss', 'business'],
        ['assistants'] ]
    """
```

For each of the word in the given parameter (separated by commas), the *digest( )* function needs to use *countRepeatingChar( )* to find the highest number of the "repeating letter" for this word. Use this number-1 as index to the list, to either build the inner-list or append the word into the inner-list.

(15 marks)

Question 4 (25 marks)

This question covers materials up to seminar 4. **The data structure to use is list. You can use more than ONE list or nested list. No dictionary collection is allowed for this question. Keep the program modular by defining necessary functions.**

Read both all parts before attempting the question. You can submit all parts as one program.

In Singapore, a Certificate of Entitlement (COE) gives you the right to own and use a vehicle in Singapore. To register a vehicle, you must first bid for a COE. You can bid for a COE during the open bidding exercises conducted twice a month. When you secure a COE from the bidding exercise, you can register a vehicle and use it for 10 years.

At each bidding exercise, bidder indicates their ID (either NRIC or FIN) and the reserve price, i.e., the maximum amount bidder is willing to pay for the COE.

Develop an application with the following menu that allows bidder to submit/update/remove their bid and run a simulation of COE allocation given a quota.

```
<< COE Menu >>
1. Submit/Update bid
2. Remove bid
3. List bids
4. Simulate allocation
0. Quit
Enter option:
```

Before presenting the menu above, you must create an empty list for storing the bid submissions.

Option 1: Submit/Update bid

- To submit/update a bid, the user must provide an ID.
- If the ID is not in the existing list of bids, this is a new bid. Ask for the reserve price and add the new bid to the end of the list.

```
Enter bidder identification: S7788223G
Enter reserve price: 96000
Your bid is submitted...
```

- If the ID can be found in the existing list of bids, your program should show the current bid's reserve price, and update the bid with the new reserve price.

```
Enter bidder identification: S9211367A
Your current reserve price is $92,034.00
Enter new reserve price: 95000
Your bid is updated...
```

- Ensure that the reserve price must be positive.

Option 2: Remove bid

- This option allows the user to remove his/her bid.
- To remove a bid, the user must provide an ID.

- If the ID cannot locate a bid, display an appropriate message.

```
Enter bidder id to remove: S3877667F
No such bidder...
```

- If the ID can be found in the existing list of bids, your program should show the current bid's reserve price, and ask for removal confirmation.

```
Enter bidder ID to remove: S3942244A
Your current reserve price is $92,010.00
Confirm remove? (Y/N): Y
Bids remove successfully
```

- Only remove the bid if the confirmation is yes.

Option 3: List bids

- No input required. This option will print all the submitted bids, with the count of bids received in the first line. No sorting required as we need to preserve the FIFO properties of the bids.
- See suggested format below:

```
Total number of bids received: 6
T6007603G  $92,001.00
F1234567D  $92,000.00
G1234567D  $92,001.00
S0945608B  $92,003.00
M1234567D  $91,900.00
S8742909A  $90,002.00
```

Option 4: Simulate allocation

- This option will run a simulation of the COE allocation exercise.
- Allow the user to enter the quota: the number of COE available for this bidding exercise, and this should be a positive integer number. Otherwise, the allocation exercise is aborted and return back to main menu.
- Using "quota", the option will start allocating COE to bidders with highest reserve price, next higher reserve price, and so on, until the quota is exhausted.
- If there are bidders with the same reserve price, your program needs to ensure FIFO properties of their bids. I.e. if there are 3 bidders with the same reserve price, and only 1 COE left to allocate, the bidder that submitted the bid first will get the allocation.
- The COE price to pay is the last successful bidder's reserve price.
- At the end of the allocation exercise, this option should print the list of successful bidders' ID, reserve price and the COE price to pay.
- Sample allocation result:

```
Enter available COE (quota): 4
S9211367A - Reserve Price $92,034.00 - COE $92,001.00
S3942244A - Reserve Price $92,010.00 - COE $92,001.00
S0945608B - Reserve Price $92,003.00 - COE $92,001.00
T6007603G - Reserve Price $92,001.00 - COE $92,001.00
```

(25 marks)

---- END OF ASSIGNMENT ----