**Question 1a)**

**Why plagiarism in coding happens**

* Students might struggle to fully understand the code they are working on, causing them to replicate it without comprehending
* Tight deadlines might lead to desperation, pressing students to plagiarize code in order to deliver their assignments on time (Rochester Institute of Technology, 2018)
* Easy access to existing codes online makes it tempting to duplicate solutions without attribution (Watters, 2011)
* Students might face intense academic pressure to earn excellent results. This pressure will cause some students to plagiarize code to achieve course requirements. (Bailey, 2020)
* Out of fear of failure or a desire for academic recognition, students may turn to submit codes that are not written by them. (Bailey, 2020)
* Some may be unaware of the ethical and legal implications of plagiarism in coding (Rochester Institute of Technology, 2018)

**Ways to avoid**

* Trainers can divide class activities into smaller parts for students to understand the fundamentals of coding
* Trainers should prioritize on teaching coding principles, encourage students to understand the fundamentals of coding
* Use plagiarism detection tools (e.g., Turnitin) to identify replicated codes (Carnegie Mellon University, n.d.)
* Create tasks which necessitate original problem-solving, limiting the probability of duplicate solutions (School of Humanities, Arts, and Social Science, n.d.)
* Provide resources such as coding tutorials, guidance sessions so that students can reach out when they encounter difficulties (Bailey, 2020)

*213 words*

**Question 1b)**

|  |
| --- |
| Age\_below\_35 **=** {"Ordinary Account":23,"Special Account":6,"MediSave Account":8}  Age\_below\_45 = {"Ordinary Account":21,"Special Account":7,"MediSave Account":8}  Age\_below\_50 = {"Ordinary Account":19,"Special Account":8,"MediSave Account":10}  Age\_below\_55 = {"Ordinary Account":15,"Special Account":11.5,"MediSave Account":10.5}  Age\_below\_60 = {"Ordinary Account":12,"Special Account":5.5,"MediSave Account":10.5}  Age\_below\_65 = {"Ordinary Account":3.5,"Special Account":4.5,"MediSave Account":10.5}  Age\_below\_70 = {"Ordinary Account":1,"Special Account":2.5, "MediSave Account":10.5}  Age\_above\_70 = {"Ordinary Account":1,"Special Account":1, "MediSave Account":10.5} |
| user\_age = int(input("What is your current age?"))  if user\_age < 35:  print(f"The percentage of CPF allocation rates for employees aged {user\_age} are {Age\_below\_35}.")  elif user\_age < 45:  print(f"The percentage of CPF allocation rates for employees aged {user\_age} are {Age\_below\_45}.")  elif user\_age < 50:  print(f"The percentage of CPF allocation rates for employees aged {user\_age} are {Age\_below\_50}.")  elif user\_age < 55:  print(f"The percentage of CPF allocation rates for employees aged {user\_age} are {Age\_below\_55}.")  elif user\_age < 60:  print(f"The percentage of CPF allocation rates for employees aged {user\_age} are {Age\_below\_60}.")    elif user\_age < 65:  print(f"The percentage of CPF allocation rates for employees aged {user\_age} are {Age\_below\_65}.")    elif user\_age < 70:  print(f"The percentage of CPF allocation rates for employees aged {user\_age} are {Age\_below\_70}.")  elif user\_age > 70:  print(f"The percentage of CPF allocation rates for employees aged {user\_age} are {Age\_above\_70}.") |

**Possible Output**

What is your current age? 40

The percentage of CPF allocation rates for employees aged 40 are {'Ordinary Account': 21, 'Special Account': 7, 'MediSave Account': 8}.

**About the code**

This Python code defines different dictionaries (Age\_below\_35, Age\_below\_45 etc.) that reflect CPF (Central Provident Fund) allocation rates for employees based on their age groups.

Each dictionary includes allocation rates for three types of accounts: Ordinary Account, Special Account, and MediSave Account.

The int(input()) function prompts the use to provide their current age as a whole number.

The code then uses if-elif statements to compare the user’s age against the series of age ranges.

Depending on the age which was entered, it prints the corresponding CPF allocation rates for their age group. For example, if the user is under the age of 35, the allocation rates from the dictionary “Age\_below\_35” will be printed.

This code is a straightforward method of informing employees about their CPF allocation rates based on their age, providing them with relevant information for financial planning.

In summary, it is an information display system based on conditional statements that assists users in understanding their CPF allocation rates based on their age.

*163 words*

**Question 1c)**

|  |
| --- |
| def get\_cpf\_allocation(age):  age\_ranges = {  "Age below 35": {"Ordinary Account":23, "Special Account":6, "MediSave Account":8},  "Age below 45": {"Ordinary Account":21, "Special Account":7, "MediSave Account":8},  "Age below 50": {"Ordinary Account":19, "Special Account":8, "MediSave Account":10},  "Age below 55": {"Ordinary Account":15, "Special Account":11.5, "MediSave Account":10.5},  "Age below 60": {"Ordinary Account": 12, "Special Account":5.5, "MediSave Account":10.5},  "Age below 65": {"Ordinary Account":3.5, "Special Account": 4.5, "MediSave Account":10.5},  "Age below 70": {"Ordinary Account": 1, "Special Account": 2.5, "MediSave Account":10.5},  "Age above 70": {"Ordinary Account": 1, "Special Account": 1, "MediSave Account":10.5},  } |
| for age\_range, allocation\_rates in age\_ranges.items():  if age < int(age\_range.split()[-1]):  return allocation\_rates  # Age is above the defined ranges  return None  # Prompt user for their age with error handling  try:  user\_age = int(input("What is your current age?"))  allocation\_rates = get\_cpf\_allocation(user\_age)  if allocation\_rates:  print(f"The percentage of CPF allocation rates for employees aged {user\_age} are {allocation\_rates}.")  else:  print("CPF allocation rates not defined for the provided age range.")  except ValueError:  print("Please enter a valid age as a positive integer.") |

**Possible Output**

What is your current age? 25.5

Please enter a valid age as a positive integer.

Runs code again

What is your current age? 20

The percentage of CPF allocation rates for employees aged 20 are {'Ordinary Account': 23, 'Special Account': 6, 'MediSave Account': 8}.

**About the code**

Get\_cpf allocation(age)

* This function retrieves CPF (Central Provident Fund) allocation rates based on a given age
* Within the function, there is a dictionary “age\_ranges” that defines allocation rates for the respective age groups.
* It iterates through the age ranges dictionary, comparing the provided age to the upper limit of each age group.
* When the appropriate age range is found, it returns the allocation rates as a dictionary.
* If the age provided does not fall within any of the defined ranges, it returns “None”, indicating that allocation rates for the age are not recognized.

Error Handling with “try-except”

(Python Programming and Numerical Methods: A Guide for Engineers and Scientists — Python Numerical Methods, n.d.)

* The code includes a “try-except” block to deal with potential errors
* Within the try block, it attempts to convert user input to an integer by using “int(input())” function
* If the conversion is successful, the program proceeds to retrieve allocation rates
* If the conversion fails (e.g. non-integer input) it raises a “ValueError”, which will be detected by the “except” block.
* In the event of a “ValueError”, it prints an error message and prompts the user to enter a valid age.

Output Presentation

* Depending on the validity of the user’s input and the age ranges defined, the code produces a suitable output
* When valid allocation rates have been obtained using the “get\_cpf\_allocation” function, it prints the respective allocation rates.
* If the age provided does not match any defined range, it informs the user that allocation rates for that age range are not defined

Enhanced Readability

* The names of variables and functions are intended to be informative and self-explanatory, enhancing the comprehension of code
* Comments are included to explain the purpose of key code sections

This revised code structure provides more versatility, error handling, clear user feedback, and adherence to coding standards, making it a flexible and user-friendly tool while avoiding plagiarism and holding onto academic integrity.

*301 words*

**Question 2**

|  |
| --- |
| def is\_valid\_product(item, products):  return item in products  def get\_item\_price():  while True:  try:  return float(input("How much is it (in SGD)? "))  except ValueError:  print("Invalid input. Please enter a valid price.")  def main():  products = ['laptop', 'mouse', 'webcam', 'keyboard', 'speaker']  updated\_items = []  print(f'We have a list of products here: {products}.')    while True:  item = input("Hello! What do you want to buy?")  if not is\_valid\_product(item, products):  print("Wrong product! Please try again.")  continue  price\_of\_item = get\_item\_price()  entered\_input = [item, price\_of\_item]  updated\_items.append(entered\_input)  query = input("Would you like to continue? (yes/no)")  if query.lower() == 'no':  print("Thank you & have a nice day!")    # Exit the loop when the user says "no"  break    elif query.lower() != 'yes':  print("Invalid response. Please enter 'yes' or 'no'.")  print(f"This is our updated shopping list: {updated\_items}")  if \_\_name\_\_ == "\_\_main\_\_":  main() |

**About the code**

Error handling is included in the modified code for validating the price input. It ensures that the application handles incorrect input and guides the user to enter correct input. The handling of errors increases the code's reliability. If a user entered an invalid price in the original code, it would result in an unattended error. The modified code shows a user-friendly error notice and requests valid input, strengthening the program's overall dependability.

The modified code includes comments which clarify the purpose of various functions and code blocks. This enhances code readability and makes the code easier to understand for future use. Code comments serve as documentation, facilitating future modifications. They also assist others in comprehending the logic and functionality of the code. The original code lacks comments which make it hard to comprehend.

The modified code modularizes several components of the program, making it more organized and readable. There are different functions for validating input and handling user requests. Modularization divides the code into multiple areas, making it easier to comprehend and maintain. Each function serves a distinct purpose, which improves code quality and reusability. The original code, on the other hand, is monolithic, making it tricky to manage as it grows.

*202 words*

**Reference(s):**

Bailey, J. (2020). Why do students commit plagiarism? *Plagiarism Today*.

<https://www.plagiarismtoday.com/2020/07/09/why-do-students-commit-plagiarism/>

Carnegie Mellon University. (n.d.). *How to prevent plagiarism - Eberly Center - Carnegie*

*Mellon University*.

<https://www.cmu.edu/teaching/designteach/teach/instructionalstrategies/writing/preventplagiarism.html>

*CPFB | CPF Contribution Changes from 1 September 2023 and 1 January 2024*. (n.d.).

<https://www.cpf.gov.sg/employer/infohub/news/cpf-related-announcements/new-contribution-rates>

*Python Programming And Numerical Methods: A Guide For Engineers And Scientists —*

*Python Numerical Methods*. (n.d.).

<https://pythonnumericalmethods.berkeley.edu/notebooks/Index.html>

Rochester Institute of Technology. (2018). *Reasons Students plagiarize or cheat | Academic*

*Integrity | RIT*.

<https://www.rit.edu/academicintegrity/reasons-students-plagiarize-or-cheat#:~:text=Reasons%20range%20from%20the%20more,Fear%20of%20failing>

School of Humanities, Arts, and Social Science. (n.d.). *Resources for Teachers: How to*

*Prevent Plagiarism*. Retrieved September 14, 2023, from

<https://cmsw.mit.edu/writing-and-communication-center/resources/teachers/prevent-plagiarism/>

Watters, A. (2011, May 2). *How the Internet Affects Plagiarism | KQED*. KQED.

<https://www.kqed.org/mindshift/11120/how-the-internet-affects-plagiarism>