



MoSCoW Requirements Prioritization

Understanding MoSCoW Requirements Prioritization



Introduction

- What is MoSCoW?
- A simple method for prioritizing project requirements
- Used in Agile, software development, and business analysis



1. Non-negotiable
2. Minimum viable product
3. Unable to deliver the end product without this
4. Not legal with it
5. Unsafe without it
6. Without this project is not viable



1. Important but not vital
2. Maybe painful to leave out but the solution is still viable
3. May need some kind of workaround



1. Desirable but not as important as Should Have
2. Only do if there is extra time and budget



1. Won't have this time around at all
2. Out of budget
3. Nice to have but has no real impact

What Does MoSCoW Stand For?

- **M**ust Have: Essential features required for project success
- **S**hould Have: Important but not critical requirements
- **C**ould Have: Nice-to-have features that add value
- **W**on't Have (this time): Features that can be deferred

Why Use MoSCoW?



- Helps teams focus on the most critical requirements
- Ensures efficient resource allocation
- Supports Agile and iterative development

Example Scenario

Developing a Mobile Banking App



Example Scenario

Developing a Mobile Banking App

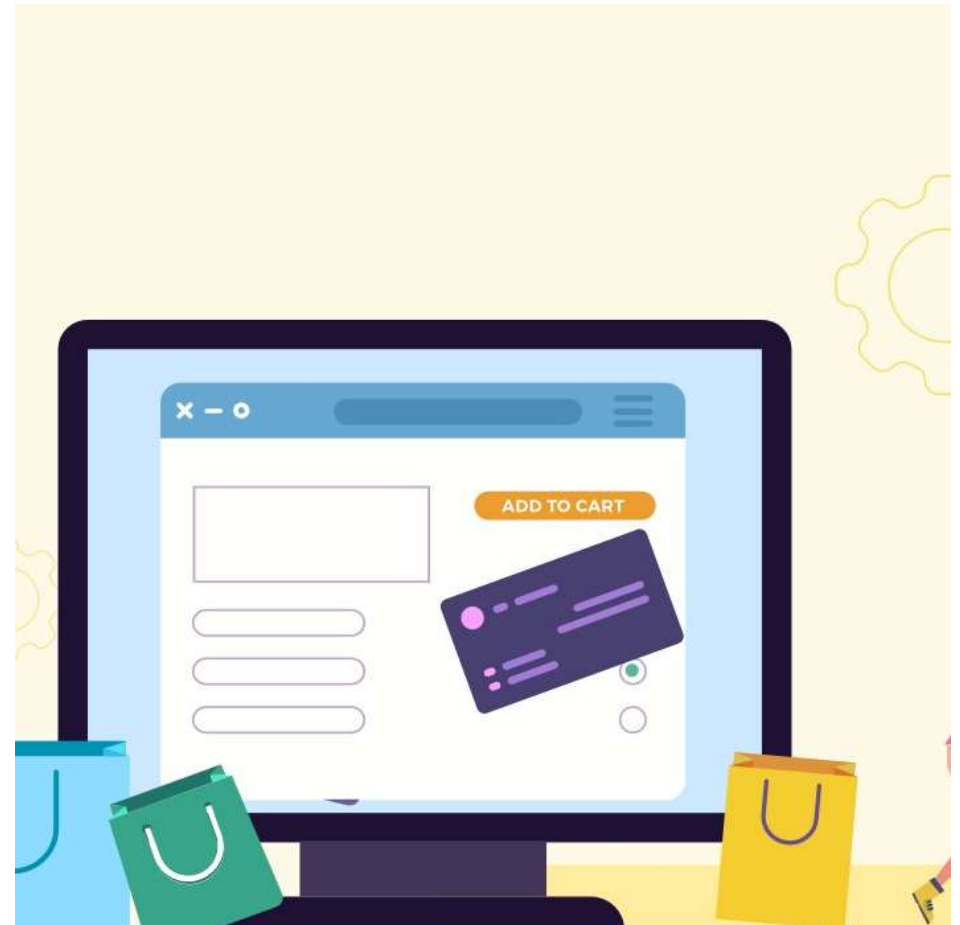
- **Must Have:** Secure login, fund transfers, account balance check
- **Should Have:** Biometric authentication, scheduled payments
- **Could Have:** Dark mode, customizable UI
- **Won't Have:** Cryptocurrency support (for this phase)



Example Scenario

E-commerce Website

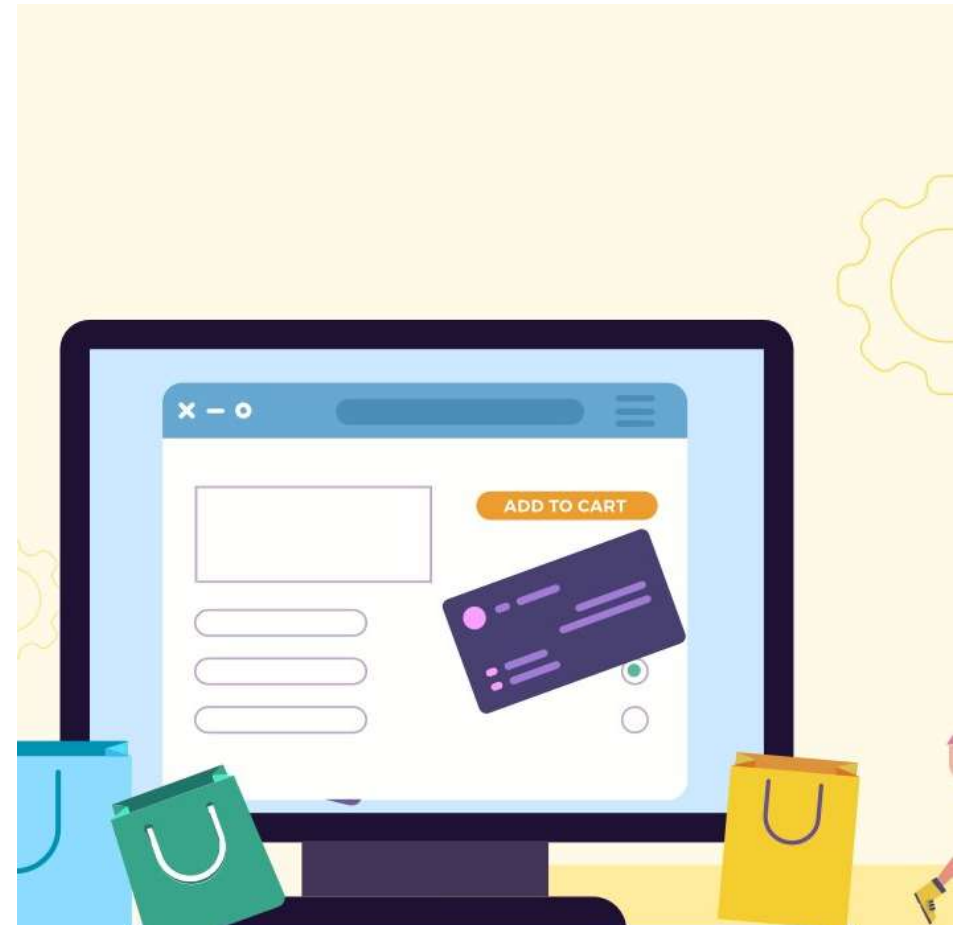
- **Must have:**
- **Should have :**
- **Could have :**
- **Won't have :**



Example Scenario

E-commerce Website

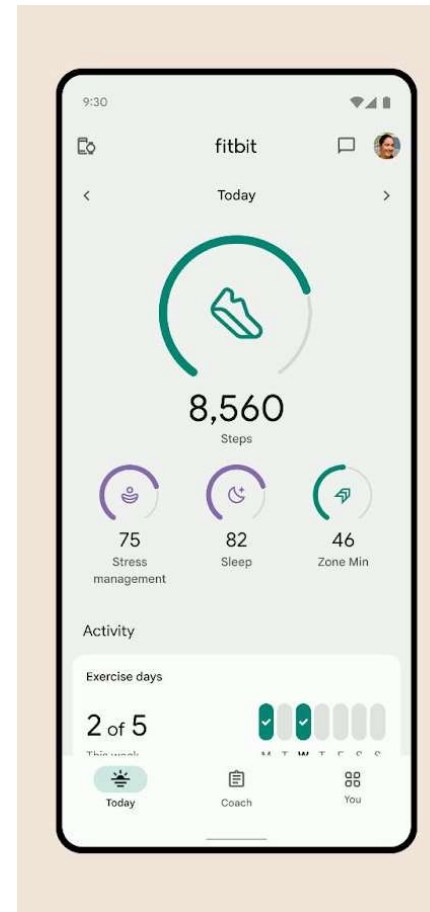
- **Must:** Secure payment gateway.
- **Should:** Product reviews section.
- **Could:** Wishlist feature.
- **Won't:** Augmented reality try-on.



Example Scenario

Fitness App

- Must have:
- Should have :
- Could have :
- Won't have :



Example Scenario- Fitness App

- **Must have:**

User registration/login (email/password).

Basic workout tracking (e.g., log exercises, duration, calories burned).

Preloaded workout plans (e.g., beginner, intermediate).

Progress dashboard (daily/weekly stats).

- **Should have :**

Integration with wearable devices (e.g., Fitbit, Apple Watch).

Customizable workout plans (user can adjust reps/sets).

Push notifications (reminders to exercise).

Social sharing (post progress to Facebook/Instagram).

- **Could have :**

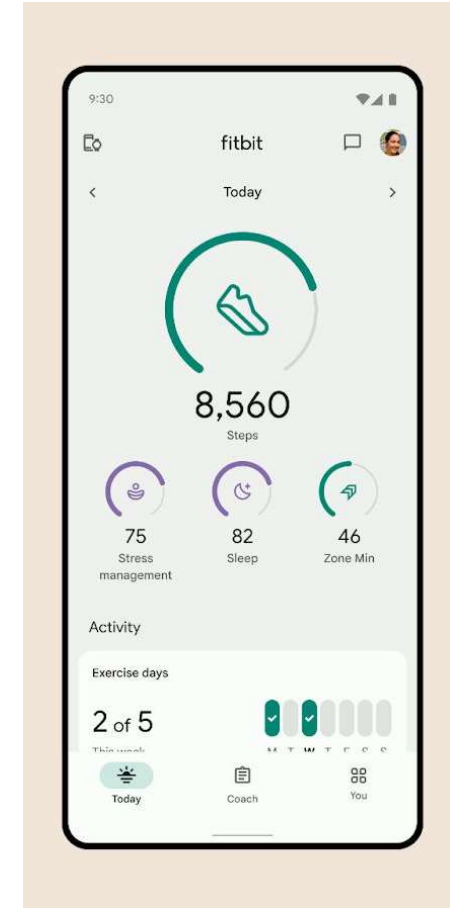
Video tutorials for exercises.

Meal-planning suggestions.

- **Won't have :**

AI-powered personalized coaching.

Live virtual classes with trainers.



Best Practices

- Not Including too many **Must Haves**
- Revisiting priorities during development
- Work with stakeholders to define priorities
- Be flexible—requirements may change
- Align MoSCoW with business goals

Quiz – Test Your Understanding

- **What does the 'M' in MoSCoW stand for?**
 - a) Mandatory
 - b) Must Have
 - c) Major Requirement
 - d) Most Important

Quiz – Test Your Understanding

- **Which of the following best describes ‘Won’t Have’ requirements?**
 - a) Features that are unnecessary and never needed
 - b) Features that are optional but will be included in the current phase
 - c) Features that will not be included in the current phase but may be considered in the future
 - d) Features that must be delivered for the project to be considered successful

Quiz – Test Your Understanding

- **Why is the MoSCoW method useful in Agile development?**
 - a) It ensures that every requested feature is implemented immediately
 - b) It helps prioritize features and manage scope within time and budget constraints
 - c) It allows customers to make all the decisions about requirements without input from developers
 - d) It eliminates the need for stakeholder discussions about priorities

Quiz – Test Your Understanding

- **What is a key risk of marking too many requirements as 'Must Have'?**
 - a) The project becomes too flexible and unstructured
 - b) The development team has more freedom to choose what to work on
 - c) It may lead to project delays and failure due to resource constraints
 - d) It makes project delivery faster and more efficient

Quiz – Test Your Understanding

- **Which of the following should be classified as a 'Should Have' requirement for an e-commerce website?**
 - a) Secure payment processing
 - b) User login and authentication
 - c) Customer reviews and ratings
 - d) Adding products to a cart

Quiz – Test Your Understanding

- **A team is developing a healthcare mobile app. Which feature should be categorized as a 'Must Have'?**
 - a) Dark mode for better user experience
 - b) Emergency contact integration for critical medical alerts
 - c) Gamification elements for user engagement
 - d) Customizable themes for user interface

Quiz – Test Your Understanding

- **A company is launching a new food delivery app. Which feature is most likely a 'Could Have'?**
 - a) Order tracking in real-time
 - b) Secure payment processing
 - c) Integration with smartwatches for order notifications
 - d) Ability to browse restaurant menus

Quiz – Test Your Understanding

- **In a university's Learning Management System (LMS), which of the following is a 'Won't Have' requirement for the initial launch?**
 - a) Online exam functionality
 - b) AI-powered personalized learning recommendations
 - c) Course material downloads
 - d) Student-teacher messaging system

Quiz – Test Your Understanding

- **A bank is developing its online banking portal. Which requirement should be categorized as 'Must Have'?**
 - a) Live chat support for customer service
 - b) Biometric login authentication
 - c) Account balance viewing and fund transfer capabilities
 - d) Personalized dashboard themes

Quiz – Test Your Understanding

- **A travel booking website is being developed. Which feature would most likely be a 'Should Have'?**
 - a) Basic flight and hotel booking functionality
 - b) Currency conversion tool for international travelers
 - c) Secure payment processing
 - d) Booking confirmation emails

Inclass Activity – MoSCoW Prioritization

- **Scenario 1:** A startup wants to build a food delivery app in 4 months.
- **Scenario 2:** A university wants to launch an online course platform in 6 months.
- **Scenario 3:** A tech company is building a voice-controlled smart home device.
- **Scenario 4:** A travel agency wants a website to compete with Booking.com.
- **Scenario 5:** A team is creating a social app for pet owners to share photos.

1. List 10–15 features.
2. Prioritize using MoSCoW.
3. **Answer the Reflection Questions**

Key Takeaways

<https://forms.office.com/e/ezX7gmmVs1>

- MoSCoW clarifies priorities and manages expectations.
- Flexibility is key—revisit categories as projects evolve.



Practice: Instructions

- Equipment or materials required – None
- The Candidates are not permitted to bring dictionaries; textbooks/case studies or any other reference materials into the exam room.
- The Candidates are only permitted to use calculators.
- Show all your calculations in your exam book
- Answer all five questions

<https://forms.office.com/e/9hBEYbYKVd>



Question 1a) – IPO charts (20 marks)

What is expected?

Input Identification - identifies all necessary inputs and describes them accurately.

Process Description - describes all processing steps with clear and explanations.

Output Specification - specifies all outputs and describes them accurately

Question 1a)

a) Create a IPO chart that will help a shop to calculate the total cost in a "20% off for purchases over £100" promotion. It should prompt for the prices of items being purchased by the customer. The program will then calculate the total cost before discount, determine if the discount applies, and display the final cost to pay. Complete an IPO Chart for the problem.

Input	Processes	Output

Question 1a - Sample solution

Input	Process	Output
Price of each item being purchased (one at a time)	1. Initialize a variable for total cost before discount (total_cost).	Total cost before discount
	2. Initialize a counter variable for the number of items (item_count).	Determine if discount applies
	3. Prompt for the price of each item, one at a time.	Final cost to pay
	4. Add the price to total_cost and increment item_count.	
	5. Repeat steps 3 and 4 until there are no more items.	
	6. If total_cost is greater than £100, apply a 20% discount.	
	7. Calculate the final cost after applying the discount (if applicable).	
	8. Display the total cost before discount, determine if the discount applies, and the final cost to pay.	

Question 1b)

Create IPO Chart that accepts the number of goals scored by a team in five World Cup matches and calculates the total goals, average goals per match, highest number of goals in a single match, and lowest number of goals in a single match. The program must display the total goals, average goals, highest goals, and lowest goals.

Input	Processes	Output

Question 1b)

Input	Process	Output
Number of goals scored in five matches	1. Prompt for the number of goals scored in each of the five World Cup matches.	Total number of goals scored
	2. Initialize variables: total_goals, highest_goals, lowest_goals.	Average number of goals per match
	3. For each match input:	Highest number of goals scored in a single match
	a. Add the goals to total_goals.	Lowest number of goals scored in a single match
	b. Update highest_goals if the current match goals are higher than the current highest_goals.	
	c. Update lowest_goals if the current match goals are lower than the current lowest_goals.	
	4. Calculate the average goals.	
	5. Display the total goals, average goals per match, highest goals, and lowest goals.	

QUESTION 2

Identify and create variables with their respective data types from the IPO charts for the above scenarios in question 1(a) and 1(b) (marks 15).

What is expected?

Identification of all variables for appropriate input handling

Accurately process the input data and store them in the respective variable

Display the required output after rprocessing

QUESTION 2

a) Create IPO chart that will help a shop to calculate the total cost in a "20% off for purchases over £100" promotion. It should prompt for the prices of items being purchased by the customer. The program will then calculate the total cost before discount, determine if the discount applies, and display the final cost to pay. Complete an IPO Chart for the problem.

QUESTION 2

Variable	Datatype
price: This variable will store the price of each item being purchased.	real numbers that have a fractional part.
total_cost: This variable will store the cumulative total cost of all items before any discount is applied.	real numbers that have a fractional part.
discount: This variable will store the discount amount if the total cost is over £100.	real numbers that have a fractional part.
final_cost: This variable will store the final cost after applying any applicable discount.	real numbers that have a fractional part.
item_count: This variable will keep track of the number of items entered.	Integers that do not have a fractional part.
more_items: This variable will store the user response to whether there are more items to be entered.	string (short for string of characters) is a data type used to represent sequences of characters,

QUESTION 2

b) Create IPO Chart that accepts the number of goals scored by a team in five World Cup matches and calculates the total goals, average goals per match, highest number of goals in a single match, and lowest number of goals in a single match. The program must display the total goals, average goals, highest goals, and lowest goals.

QUESTION 2

Variables and Data Types:

- goals_match1, goals_match2, goals_match3, goals_match4, goals_match5: Integer variables to store goals scored in each match.
- total_goals: Integer variable to store the sum of goals across all matches.
- average_goals: Float variable to store the average number of goals per match.
- highest_goals, lowest_goals: Integer variables to store the maximum and minimum goals scored in any single match.

QUESTION 3

Write pseudocode for the following problem (marks 20):

Design a solution to calculate the total amount to pay a delivery driver based on the number of deliveries completed in a week. The Delivery Manager wants to compute the payment including a fixed rate per delivery and an additional bonus based on the number of deliveries. The solution should include:

The total number of deliveries completed in a week.

The Delivery Driver receives a fixed payment of £5.50 per delivery.

The Driver also receives a bonus of £20 if they complete more than 10 deliveries in a week.

The program should output:

The total number of deliveries completed.

The total earnings from fixed payments (based on deliveries).

The bonus received, if applicable.

The total payment (sum of fixed payments and bonus) that the driver will receive.

Produce possible pseudocode for the given scenario.

QUESTION 3

What is expected?

Criteria	Description
Initialization of Variables	Variables (total_deliveries, fixed_rate_per_delivery, bonus_threshold, bonus_amount, total_earnings, bonus_received, total_payment) are correctly initialized.
Input Processing	Input total_deliveries_completed is correctly processed and assigned to total_deliveries.
Calculation of Total Earnings	Calculates total_earnings correctly as $\text{total_deliveries} * \text{fixed_rate_per_delivery}$.
Bonus Calculation and Condition	Checks if $\text{total_deliveries} > \text{bonus_threshold}$ and correctly sets bonus_received to bonus_amount if true.
Calculation of Total Payment	Calculates total_payment correctly as the sum of total_earnings and bonus_received.
Output Statements	Outputs total_deliveries, total_earnings, bonus_received, and total_payment in the correct format.
Clarity and Readability	Pseudocode is written clearly, using understandable variable names and logical structure.
Correctness of Logic	Logic flow ensures accurate computation of payments based on deliveries and bonus criteria specified.

QUESTION 3

1. Start

2. Initialize variables:

```
total_deliveries = 0
```

```
fixed_rate_per_delivery = 5.50
```

```
bonus_threshold = 10 // number of deliveries needed to qualify for bonus
```

```
bonus_amount = 20.00
```

```
total_earnings = 0.00
```

```
bonus_received = 0.00
```

```
total_payment = 0.00
```

QUESTION 3

3. Input `total_deliveries_completed`
4. Set `total_deliveries` to `total_deliveries_completed`
5. Calculate `total_earnings` as:
$$\text{total_earnings} = \text{total_deliveries} * \text{fixed_rate_per_delivery}$$
6. If `total_deliveries > bonus_threshold`:
 - 6.1. Set `bonus_received` to `bonus_amount`
 - 6.2. Add `bonus_amount` to `total_payment`
7. Set `total_payment` to `total_earnings + bonus_received`

QUESTION 3

8. Output `total_deliveries`
9. Output `total_earnings`
10. Output `bonus_received`
11. Output `total_payment`
12. End

QUESTION 4



Prifysgol Cymru
Y Drindod Dewi Sant
University of Wales
Trinity Saint David

Trawsnewid Addysg; Trawsnewid Bywydau
Transforming Education; Transforming Lives

For the above scenario (Question 3) interpret variable values in a pseudocode via trace tables (marks 20).

Replace the variable names as you have created in the pseudocode in question 3. It is not necessary to use all the columns.

You can add/remove the rows and columns
(use according to your pseudocode solution).

Consider the total number of deliveries completed in week to be **1, 6, and 13** for the trace table.

Line of pseudocode	Variable 1	Variable 2	Variable 3	Variable 4	Variable 5	Variable 6
1						
2						
3						
4						
5						
6						
7						

Line of pseudocode	total_deliveries	fixed_rate_per_delivery	bonus_threshold	bonus_amount	total_earnings	bonus_received	total_payment
1. Start	-	-	-	-	-	-	-
2. Initialize variables	-	5.50	10	20.00	-	-	-
3. Input total_deliveries_completed	1	-	-	-	-	-	-
4. Set total_deliveries	1	-	-	-	-	-	-
5. Calculate total_earnings	$1 * 5.50 = 5.50$	-	-	-	5.50	-	-
6. If total_deliveries > bonus_threshold	False	-	-	-	-	-	-
7. Set total_payment	-	-	-	-	-	-	-
8. Output	1	5.50	-	-	5.50	-	5.50
12. End	-	-	-	-	-	-	-

Line of pseudocode	total_deliveries	fixed_rate_per_delivery	bonus_threshold	bonus_amount	total_earnings	bonus_received	total_payment
1. Start	-	-	-	-	-	-	-
2. Initialize variables	-	5.50	10	20.00	-	-	-
3. Input total_deliveries_completed	6	-	-	-	-	-	-
4. Set total_deliveries	6	-	-	-	-	-	-
5. Calculate total_earnings	$6 * 5.50 = 33.00$	-	-	-	33.00	-	-
6. If total_deliveries > bonus_threshold	False	-	-	-	-	-	-
7. Set total_payment	-	-	-	-	-	-	-
8. Output	6	5.50	-	-	33.00	-	33.00
12. End	-	-	-	-	-	-	-



Line of pseudocode	total_deliveries	fixed_rate_per_delivery	bonus_threshold	bonus_amount	total_earnings	bonus_received	total_payment
1. Start	-	-	-	-	-	-	-
2. Initialize variables	-	5.50	10	20.00	-	-	-
3. Input total_deliveries_completed	13	-	-	-	-	-	-
4. Set total_deliveries	13	-	-	-	-	-	-
5. Calculate total_earnings	13 * 5.50 = 71.50	-	-	-	71.50	-	-
6. If total_deliveries > bonus_threshold	True	-	-	-	-	-	-
7. Set bonus_received	-	-	-	20.00	-	20.00	-
8. Set total_payment	-	-	-	-	-	-	-
12. End	-	-	-	-	-	-	-



QUESTION 5



Prifysgol Cymru
Y Drindod Dewi Sant
University of Wales
Trinity Saint David

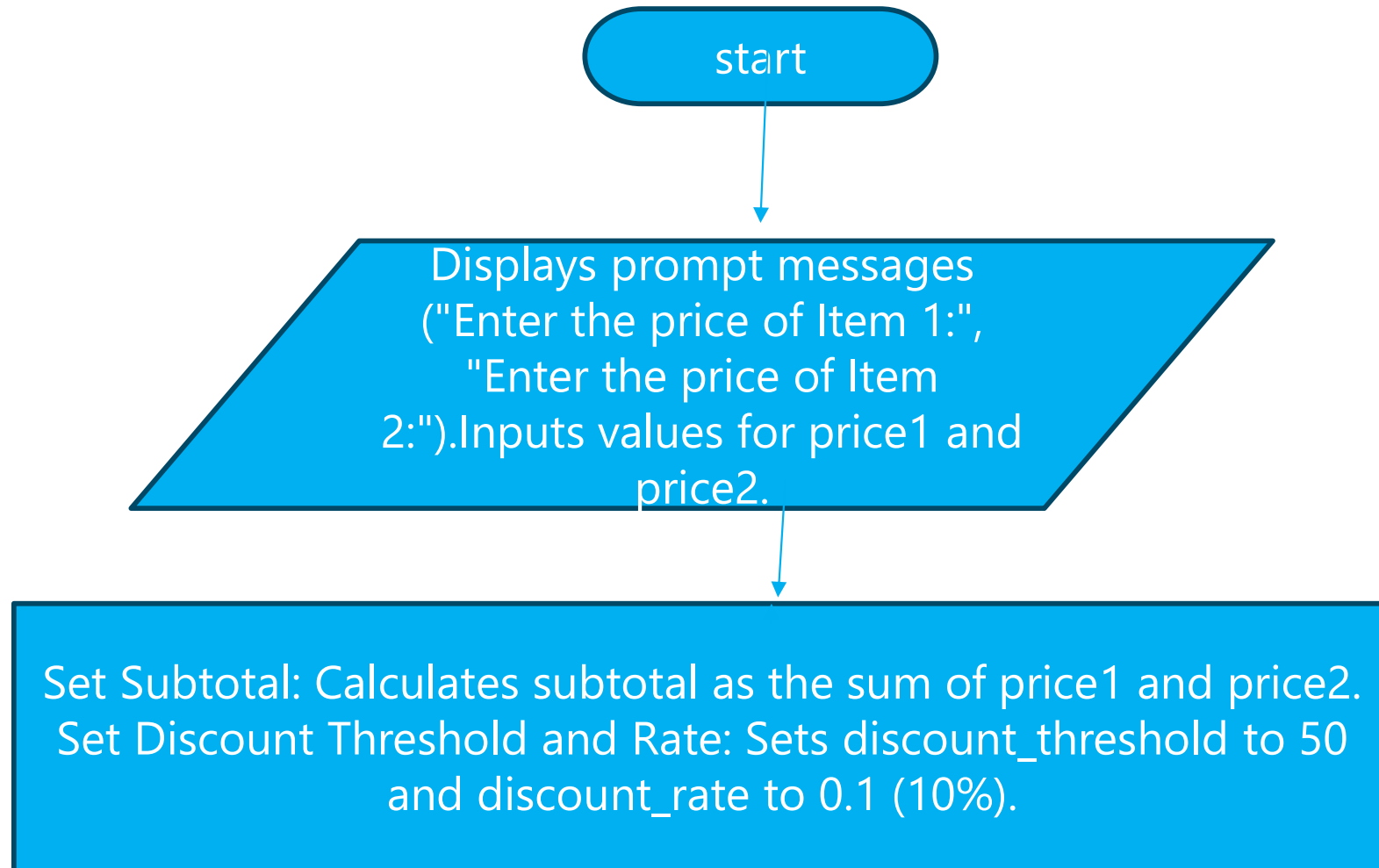
Trawsnewid Addysg; Trawsnewid Bywydau
Transforming Education; Transforming Lives

For the given pseudocode construct a flowchart to illustrate the logical program flow (marks 25).

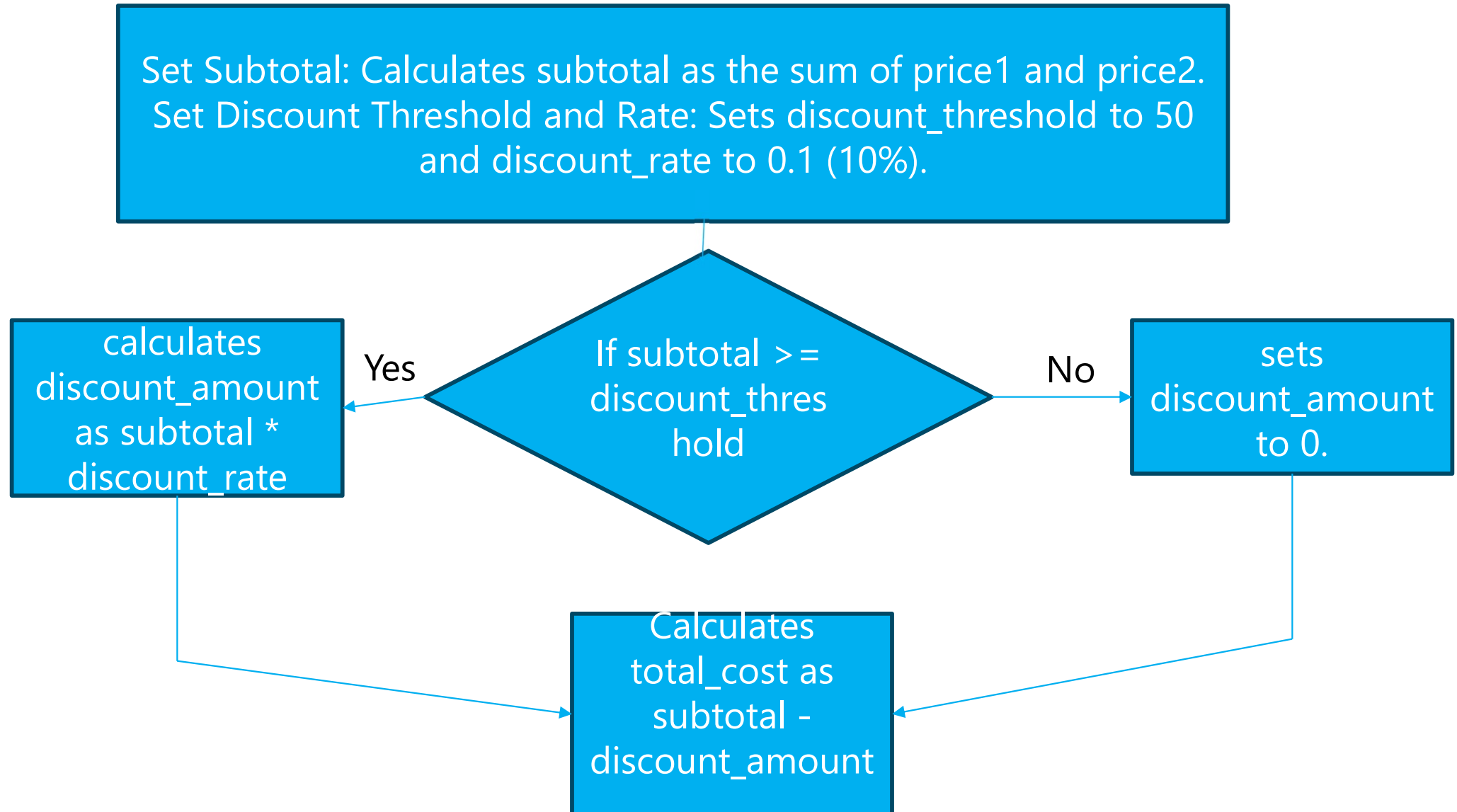
Pseudocode:

1. Display "Enter the price of Item 1:"
2. Input price1
3. Display "Enter the price of Item 2:"
4. Input price2
5. Set subtotal = price1 + price2
7. Set discount_threshold = 50
8. Set discount_rate = 0.1 // 10% discount if subtotal $\geq$ £50
9. If subtotal $\geq$ discount_threshold then
10. Set discount_amount = subtotal * discount_rate
11. Else
12. Set discount_amount = 0
13. End If
14. Set total_cost = subtotal - discount_amount
15. Display "Subtotal: £", subtotal
16. Display "Discount applied: £", discount_amount
17. Display "Total cost: £", total_cost

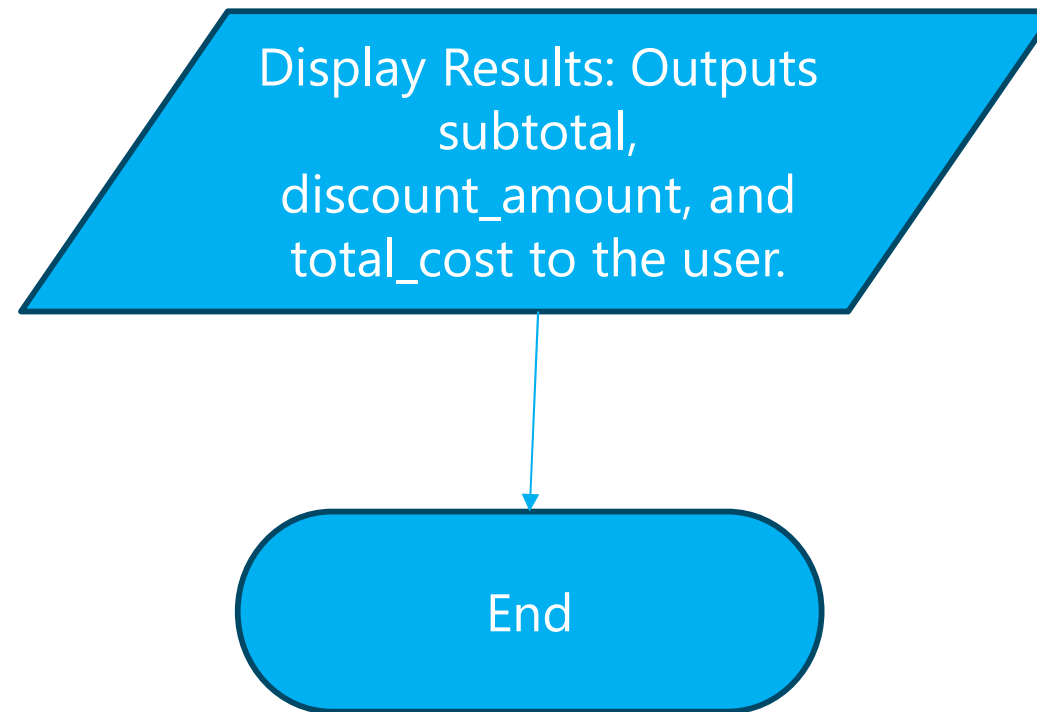
For the given pseudocode construct a flowchart to illustrate the logical program flow (marks 25).



For the given pseudocode construct a flowchart to illustrate the logical program flow (marks 25).



For the given pseudocode construct a flowchart to illustrate the logical program flow (marks 25).



Lab Exercise

- Download instructions from Moodle