ITS100 Lab Final Quiz

# Q1

Write a program to calculate the transaction balance of a student’s bank account.

First, the program takes an integer input (*b*) for the *initial balance* of the student’s bank account. After that, the program repeatedly takes two inputs (*x* and *y*) for a *transaction type* and *amount*. Notes that there are only two valid transaction types: D (deposit) and W (withdrawal). The transaction amount is numeric. If the transaction type is a deposit and the amount is numeric, then display the deposit amount and current balance. Otherwise, if the transaction type is a withdrawal, the amount is numeric, and the current balance is greater than withdrawal amount, then display the withdrawal amount and current balance. Otherwise, print (Invalid transaction!) to alert the user. The program repeats until the user inputs “**done 0**”. Then the program will display the current balance.

**Example 1** (user inputs are in **bold**)

Enter the initial balance: **1000**

Transaction type and amount ("done 0" to exit): **done 0**

* Current balance = 1000 THB.

**Example 2** (user inputs are in **bold**)

Enter the initial balance: **1000**

Transaction type and amount ("done 0" to exit): **D 500**

* Deposit = 500 THB, current balance = 1500 THB. Transaction type and amount ("done 0" to exit): **W 800**
* Withdrawal = 800 THB, current balance = 700 THB. Transaction type and amount ("done 0" to exit): **D 400**
* Deposit = 400 THB, current balance = 1100 THB. Transaction type and amount ("done 0" to exit): **done 0**
* Current balance = 1100 THB.

**Example 3** (user inputs are in **bold**)

Enter the initial balance: **1000**

* Current balance = 1000 THB.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Transaction type and amount | ("done | 0" | to | exit): | **K** | **100** |
| * Invalid transaction! |  |  |  |  |  |  |
| Transaction type and amount | ("done | 0" | to | exit): | **D** | **-200** |
| * Invalid transaction! |  |  |  |  |  |  |
| Transaction type and amount | ("done | 0" | to | exit): | **W** | **1200** |
| * Invalid transaction! |  |  |  |  |  |  |
| Transaction type and amount | ("done | 0" | to | exit): | **done 0** | |

## Test cases for Q1

**Test 1** (user inputs are in **bold**)

Enter the initial balance: **1000**

Transaction type and amount ("done 0" to exit): **done 0**

* Current balance = 1000 THB.

**Test 2** (user inputs are in **bold**)

Enter the initial balance: **1000**

Transaction type and amount ("done 0" to exit): **D 500**

* Deposit = 500 THB, current balance = 1500 THB. Transaction type and amount ("done 0" to exit): **W 800**
* Withdrawal = 800 THB, current balance = 700 THB. Transaction type and amount ("done 0" to exit): **D 400**
* Deposit = 400 THB, current balance = 1100 THB. Transaction type and amount ("done 0" to exit): **done 0**
* Current balance = 1100 THB.

**Test 3** (user inputs are in **bold**)

Enter the initial balance: **1000**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Transaction type and amount | ("done | 0" | to | exit): | **K** | **100** |
| * Invalid transaction!   Transaction type and amount | ("done | 0" | to | exit): | **D** | **-200** |
| * Invalid transaction!   Transaction type and amount | ("done | 0" | to | exit): | **W** | **1200** |
| * Invalid transaction! |  |  |  |  |  |  |
| Transaction type and amount | ("done | 0" | to | exit): | **done 0** | |
| * Current balance = 1000 THB. | | | | |  |  |
| **Test 4** (user inputs are in **bold**) | | | | |  |  |
| Enter the initial balance: **100**  Transaction type and amount ("done 0" to exit): | | | | | **W** | **200** |
| * Invalid transaction!   Transaction type and amount ("done 0" to exit): | | | | | **D** | **500** |
| * Deposit = 500 THB, current balance = 600 THB.   Transaction type and amount ("done 0" to exit): | | | | | **W** | **200** |
| * Withdrawal = 200 THB, current balance = 400 THB. Transaction type and amount ("done 0" to exit): **W -200** * Invalid transaction!   Transaction type and amount ("done 0" to exit): **done 100**   * Invalid transaction!   Transaction type and amount ("done 0" to exit): **done 0**   * Current balance = 400 THB. | | | | | | |

ITS100 Lab Final Quiz

# Q2

Write a program to repeatedly take two inputs for *student name* and *score* (i.e., 0 – 100) until the user inputs “**end 0**”. For example, the user inputs “Nancy 50” for the student has name Nancy and score 50 points. If the name is already taken once by the program, then display the message “Duplicated name!”. Otherwise, if the score is less than 0 or greater than 100, then display the message “Invalid score!”. Otherwise, the program will keep student names and scores in lists. The program will display the list of students and scores sequenced from the highest score to the lowest score. If there is no record, then display the message (> empty list!). The program must have at least 1 function (def).

**Example 1** (user inputs are in **bold**)

Enter student name and score: **Nancy 50** Enter student name and score: **James 60** Enter student name and score: **Winny 75** Enter student name and score: **Dandy 45** Enter student name and score: **Ammy 55** Enter student name and score: **end 0** List:

Winny 75

James 60

Ammy 55

Nancy 50

Dandy 45

**Example 2** (user inputs are in **bold**)

Enter student name and score: **end**

List:

* empty list!

**Example 3** (user inputs are in **bold**)

Enter student name and score: **Nan 60** Enter student name and score: **Jan 50** Enter student name and score: **Win 70** Enter student name and score: **Jan 60** Duplicate name!

Enter student name and score: **Boy 120**

Invalid score!

Enter student name and score: **Boy 60** Enter student name and score: **end 0** List:

Win 70

Nan 60

Boy 60

Jan 50

## Test cases for Q4

**Test 1** (user inputs are in **bold**)

Enter student name and score: **Nancy 50** Enter student name and score: **James 60** Enter student name and score: **Winny 75** Enter student name and score: **Dandy 45** Enter student name and score: **Ammy 55** Enter student name and score: **end 0** List:

Winny 75

James 60

Ammy 55

Nancy 50

Dandy 45

**Test 2** (user inputs are in **bold**)

Enter student name and score: **end**

List:

* empty list!

**Test 3** (user inputs are in **bold**)

Enter student name and score: **Nan 60** Enter student name and score: **Jan 50** Enter student name and score: **Win 70** Enter student name and score: **Jan 60** Duplicate name!

Enter student name and score: **Boy 120**

Invalid score!

Enter student name and score: **Boy 60** Enter student name and score: **end 0** List:

Win 70

Nan 60

Boy 60

Jan 50

**Test 4** (user inputs are in **bold**)

Enter student name and score: **Nan 101**

Invalid score!

Enter student name and score: **Boy -1**

Invalid score!

Enter student name and score: **end 0**

List:

* empty list!

ITS100 Lab Final Quiz

# Q3

Write a program that manages *reward points* of a convenient store for a user. The user is initially given 100 points. For each 50 Baht the user spends at the store, one point is rewarded. Each point is equal to 1 Baht.

The program repeatedly accepts commands from the user. A command can be either P (when the user pays money to earn points), and R (when the user redeems points) followed by a positive integer value. When the user inputs a P command, the program calculates the earned points, accumulates the points, and shows the current accumulated points. When the user inputs an R command, the program deducts the points and displays the current accumulated points. The program alerts "*Not enough points*" when the points that the user tries to redeem is greater than the accumulated points. The program alerts "*Invalid command*" when the user inputs a command other than P and R, or the user inputs a non-numeric or negative value for the amount. The program stops when the user inputs "done 0".

**Example 1** (user inputs are in *italics*)

Command: *P 101*

You earned 2 points

Your accumulated points = 102 points Command: *R 1200*

Not enough points Command: *R 10*

You redeemed 10 points

Your accumulated points = 92 points Command: *P 523*

You earned 10 points

Your accumulated points = 102 points Command: *R 52*

You redeemed 52 points

Your accumulated points = 50 points Command: *done 0*

**Example 2** (user inputs are in *italics*)

Command: *P -10* Invalid command Command: *A 10* Invalid command Command: *R -20* Invalid command Command: *R 10*

You redeemed 10 points

Your accumulated points = 90 points Command: *P 49*

You earned 0 points

Your accumulated points = 90 points Command: *done 0*

## Test cases for Q1

|  |  |  |
| --- | --- | --- |
| **No.** | **Input** | **Expected Output** |
| 1 | done 0 |  |
| 2 | P -10  A 10  R -20  done 0 | Invalid command Invalid command Invalid command |
| 3 | P 101  R 1200  R 10  done 0 | You earned 2 points  Your accumulated points = 102 points Not enough points  You redeemed 10 points  Your accumulated points = 92 points |
| 4 | P 100  R 100  done 0 | You earned 2 points  Your accumulated points = 102 points You redeemed 100 points  Your accumulated points = 2 points |

ITS100 Lab Final Quiz

# Q4

Write a program that repeatedly accepts players' names and scores until the user inputs "done". The program then orders the players according to their scores from the highest to the lowest. The players receive medals according to the following conditions:

* If a player's score is the highest among all players, the player receives a "Gold" medal.
* A player who does not receive the Gold medal, but his/her score is higher than the average score receives a "Silver" medal.

The program prints the list of players with their scores and medals ordered from the highest score to the lowest score. It uses a tab ("\t") to separate columns.

The program alerts "*Invalid input*" when an input score is non-numeric or negative. The program alerts "*Duplicated player*" when the user inputs a previously input name. The program prints "*No players*" when the user inputs only "done"

**Example 1** (user inputs are in *italics*)

Input: *Pete 82*

Input: *John 100*

Input: *Pete 60* Duplicated player Input: *Penny -1* Invalid input Input: *Michael xx* Invalid input Input: *Sheldon 99*

Input: *Susan 102* Input: *done* Susan 102

John 100

Sheldon 99

Gold

Silver Silver

Pete 82

**Example 2** (user inputs are in *italics*)

Input: *done*

No players

## Test cases for Q1

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No.** | **Input** | **Expected Output** | | |
| 1 | done | No | players | |
| 2 | A 10 | C | 30 | Gold |
|  | B 22 | B | 22 | Silver |
|  | C 30 | A | 10 |  |
|  | done |  |  |  |
| 3 | A 10 | Invalid input | | |
|  | A -10 | Duplicated player | | |
|  | A 20 | A 20 Gold | | |
|  | done |  | | |
| 4 | A 30 | A | 30 | Gold |
|  | B 10 | D | 26 | Silver |
|  | C 25 | C | 25 | Silver |
|  | D 26 | B | 10 |  |
|  | done |  |  |  |

ITS100 Lab Final Quiz

# Q5

Write a program to repeatedly take an integer input from a user. If the input is not between 0 and 9, then display the message (Invalid input!) to alert the user and stop the program as shown in Examples 1 and 2. Otherwise, the program will display an *isosceles triangle* shape of the number as shown as shown in Example 3. The program repeats until the user inputs “**0**” (number zero). Then display the message (Exit program. Bye!) and stop the program.

Note that the isosceles triangle is a triangle having two sides of equal length.

|  |  |
| --- | --- |
| **Example 1** (user inputs are in **bold**) |  |
| Enter an integer number (0 to exit): Valid value is between 0 and 9! | **10** |
| **Example 2** (user inputs are in **bold**) |  |
| Enter an integer number (0 to exit): 3 | **3** |
| 3 3 |  |
| 3 3 3 |  |
| Enter an integer number (0 to exit): Valid value is between 0 and 9! | **-5** |

**Example 3** (user inputs are in **bold**)

Enter an integer number (0 to exit): **1**

1

Enter an integer number (0 to exit): **2**

2

2 2

Enter an integer number (0 to exit): **3**

3

3 3

3 3 3

Enter an integer number (0 to exit): **6**

6

6 6

6 6 6

6 6 6 6

6 6 6 6 6

6 6 6 6 6 6

Enter an integer number (0 to exit): **0**

Exit program. Bye!

## Test cases for Q2

**Test 1** (user inputs are in **bold**)

Enter an integer number (0 to exit): **10**

Valid value is between 0 and 9!

**Test 2** (user inputs are in **bold**)

Enter an integer number (0 to exit): **-1**

Valid value is between 0 and 9!

**Test 3** (user inputs are in **bold**)

Enter an integer number (0 to exit): **3**

3

3 3

3 3 3

Enter an integer number (0 to exit): **-5**

Valid value is between 0 and 9!

**Test 4** (user inputs are in **bold**)

Enter an integer number (0 to exit): **1**

1

Enter an integer number (0 to exit): **2**

2

2 2

Enter an integer number (0 to exit): **3**

3

3 3

3 3 3

Enter an integer number (0 to exit): **4**

4

4 4

4 4 4

4 4 4 4

Enter an integer number (0 to exit): **5**

5

5 5

5 5 5

5 5 5 5

5 5 5 5 5

Enter an integer number (0 to exit): **6**

6

6 6

6 6 6

6 6 6 6

6 6 6 6 6

6 6 6 6 6 6

9

9 9

9 9 9

9 9 9 9

9 9 9 9 9

9 9 9 9 9 9

9 9 9 9 9 9 9

9 9 9 9 9 9 9 9

9 9 9 9 9 9 9 9 9

Enter an integer number (0 to exit): **0**

Exit program. Bye!

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Enter an integer  7 | number | (0 | to | exit): | **7** |
| 7 7 |  |  |  |  |  |
| 7 7 7 |  |  |  |  |  |
| 7 7 7 7 |  |  |  |  |  |
| 7 7 7 7 7 |  |  |  |  |  |
| 7 7 7 7 7 7 |  |  |  |  |  |
| 7 7 7 7 7 7 7 |  |  |  |  |  |
| Enter an integer 8 | number | (0 | to | exit): | **8** |
| 8 8 |  |  |  |  |  |
| 8 8 8 |  |  |  |  |  |
| 8 8 8 8 |  |  |  |  |  |
| 8 8 8 8 8 |  |  |  |  |  |
| 8 8 8 8 8 8 |  |  |  |  |  |
| 8 8 8 8 8 8 8 |  |  |  |  |  |
| 8 8 8 8 8 8 8 8 |  |  |  |  |  |
| Enter an integer | number | (0 | to | exit): | **9** |

ITS100 Lab Final Quiz

# Q6

Write a program to repeatedly take two inputs for *student ID* (4-digit number) and *score* (i.e., 0 – 100) until the user inputs “**done 0**”. For example, the user inputs “1234 60” for the student has ID 1234 and score 60 points. If the student ID is a 4-digit number and is new, and the score is between 0 and 100, then record the student ID and score in lists. Otherwise, if the student ID already recorded, the display the message (“The ID is already recorded!”). Moreover, the program should display (“Invalid ID format!”) for the ID input that not a 4-digit number, and display (“Invalid score!”) for the score that is less than 0 or greater than 100, respectively.

The program will display the list of student IDs and scores sequenced by student IDs. Then the program will also display the maximum, average, and minimum scores of students. If there is no record, then display the message (> no score is recorded!). The program must have at least 1 function (def).

**Example 1** (user inputs are in **bold**) – Hint: good inputs

There are 5 student(s). The average score is 56.00 points.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Enter | student | ID | and | score: | **1234** | **60** |
| Enter | student | ID | and | score: | **2222** | **45** |
| Enter | student | ID | and | score: | **4444** | **60** |
| Enter | student | ID | and | score: | **1111** | **40** |
| Enter | student | ID | and | score: | **3333** | **75** |
| Enter | student | ID | and | score: | **1111** | **45** |
| The ID is already recorded!  Enter student ID and score: **3333 70**  The ID is already recorded! | | | | | | |
| Enter  List: 1111 | student ID  40 | | and score: | | **done** | **0** |
| 1234 | 60 | |  | |  |  |
| 2222 | 45 | |  | |  |  |
| 3333 | 75 | |  | |  |  |
| 4444 | 60 | |  | |  |  |

**Example 2** (user inputs are in **bold**) – Hint: invalid inputs

Enter student ID and score: **123 40**

Invalid ID format!

Enter student ID and score: **12345 40**

Invalid ID format!

Enter student ID and score: **abcd 40**

Invalid ID format!

Enter student ID and score: **1234 -1**

Invalid score!

Enter student ID and score: **1234 101**

Invalid score!

Enter student ID and score: **done 0**

List:

* no score is recorded!

## Test cases for Q5

**Test 1** (user inputs are in **bold**)

There are 5 student(s). The average score is 56.00 points.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Enter | student | ID | and | score: | **1234** | **60** |
| Enter | student | ID | and | score: | **2222** | **45** |
| Enter | student | ID | and | score: | **4444** | **60** |
| Enter | student | ID | and | score: | **1111** | **40** |
| Enter | student | ID | and | score: | **3333** | **75** |
| Enter | student | ID | and | score: | **1111** | **45** |
| The ID is already recorded!  Enter student ID and score: **3333 70**  The ID is already recorded! | | | | | | |
| Enter  List: 1111 | student ID  40 | | and score: | | **done** | **0** |
| 1234 | 60 | |  | |  |  |
| 2222 | 45 | |  | |  |  |
| 3333 | 75 | |  | |  |  |
| 4444 | 60 | |  | |  |  |

**Test 2** (user inputs are in **bold**)

Enter student ID and score: **done 0**

List:

* no score is recorded!

**Test 3** (user inputs are in **bold**)

Enter student ID and score: **123 40**

Invalid ID format!

Enter student ID and score: **12345 40**

Invalid ID format!

Enter student ID and score: **abcd 40**

Invalid ID format!

Enter student ID and score: **1234 -1**

Invalid score!

Enter student ID and score: **1234 101**

Invalid score!

Enter student ID and score: **done 0**

List:

* no score is recorded!

**Test 4** (user inputs are in **bold**)

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Enter | student | ID | and | score: | **1234** | **60** | |
| Enter | student | ID | and | score: | **2345** | **55** | |
| Enter | student | ID | and | score: | **2222** | **25** | |
| Enter | student | ID | and | score: | **1122** | **65** | |
| Enter | student | ID | and | score: | **1111** | **55** | |
| Enter | student | ID | and | score: | **2222** | **50** | |
| The ID is already recorded!  Enter student ID and score: **nami 40**  Invalid ID format! | | | | | | | |
| Enter student ID and  Invalid ID format! Enter student ID and | | | | score:  score: | **333 40**  **55555 40** | | |
| Invalid ID format!  Enter student ID and | | | | score: | **4444 -1** | | |
| Invalid score!  Enter student ID and | | | | score: | **4444 101** | | |
| Invalid score!  Enter student ID and List: | | | | score: | **done 0** | | |
| 1111 | 55 | | | |  | |  |
| 1122 | 65 | | | |  | |  |
| 1234 | 60 | | | |  | |  |
| 2222 | 25 | | | |  | |  |
| 2345  There | 55  are 5 student(s). The | | | | average score | | is 52.00 points. |

ITS100 Lab Final Quiz

# Q2

Write a Python program that repeatedly accepts students’ names, their preferred study programs (either “ce”, “che”, “ec”, “ie”, or “me”) and their GPAs (between 0.00 and 4.00) until the user inputs “done”.

For example, the user inputs “tony me 3.22” means a student named “Tony” with the GPA of 3.22 prefers the ME program. The program prints “ERROR” when the user inputs a wrong study program, or a GPA is less than 0.00 or greater than 4.00.

The program then displays the number of the students in each program, as well as the list of the students sorted by their GPAs from the largest to the smallest. The GPAs must be printed using %.2f.

**Example 1** (user inputs are in *italics*)

Input: *done*

SUMMARY

ce = 0

che = 0

The list is empty.

|  |  |  |
| --- | --- | --- |
| ec | = | 0 |
| ie | = | 0 |
| me | = | 0 |
|  |  | LIST |

**Example 2** (user inputs are in *italics*)

Input: *tony me 3.22* Input: *steve ce 2.45* Input: *natasha che 3.45* Input: *scott ec 5.00* ERROR

Input: *peter mt 1.25*

ERROR

Input: *clint ie 2.21* Input: *bruce ce 2.75* Input: *done*

SUMMARY

ce = 2

che = 1

ec = 0

ie = 1

me = 1

LIST

natasha che 3.45

tony me 3.22

bruce ce 2.75

steve ce 2.45

clint ie 2.21

ITS100 Lab Final Quiz

# Q3

Write a Python program to help tally the voting results. This program repeatedly accepts integers between 1 and 1000 until the user inputs “done”. The program prints “ERROR” when the user inputs a wrong/invalid value.

The program then displays the input integers and their frequencies ordered from the smallest to the largest as shown in the following examples. The program displays “The list is empty” when the user did not input any valid value.

**Example 1** (user inputs are in *italics*)

Input: *done*

SUMMARY

The list is empty

**Example 2** (user inputs are in *italics*)

Input: *1*

Input: *5*

Input: *10* Input: *one* ERROR

Input: *1001*

ERROR

Input: *1000*

Input: *42*

Input: *5*

Input: *done*

SUMMARY

1 1

5 2

10 1

42 1

1000 1

ITS100 Lab Final Quiz

# Q7

Write a program to repeatedly take character inputs and keep the values in a list. The character is either “x” or “o”. The program repeats until the user inputs the total of 9 characters of “x” or “o”.

If the input is neither “x” nor “o”, then alert “Wrong input” to the user and stop the program. The program will then display all characters in form of XO-board after completing 9 characters.

An XO-board has 3 columns and 3 rows. The characters are displayed in order from left to right and top to bottom. The characters “x” or “o” are lowercase letters only.

**Example 1** (user inputs are in *italics*)

Input: *8*

Wrong input

**Example 2** (user inputs are in *italics*)

Input: *o* Input: *x* Input: *X* Wrong input

\*Last input from example is uppercase *“X”* character.

**Example 3** (user inputs are in *italics*)

Input: *o* Input: *x* Input: *o* Input: *o* Input: *x* Input: *x* Input: *x* Input: *o* Input: *o*

|o|x|o|

|o|x|x|

|x|o|o|

ITS100 Lab Final Quiz

# Q8

Write a program that repeatedly accepts currencies and their amount (greater than or equal to 1) until the user inputs “end”. For example, the user inputs “JPY 10000” when there is a JPY currency for 10000 YEN. The program then displays the conversions of all currencies to THB.

* The program prints “ERROR” when a currency is not in the conversion table or when the amount is less than 1.
* All values are displayed by using “%.2f” format Exchange rate conversion table

|  |  |
| --- | --- |
| **Currency** | **Exchange to Thai Bath (THB)** |
| 1 Japanese Yen (JPY) | 0.29 |
| 1 United States Dollar (USD) | 31.99 |
| 1 Euro (EUR) | 35.62 |

**Example 1** (user inputs are in *italics*)

Input: *end*

**Example 2** (user inputs are in *italics*)

Input: *JPY 5000*

Input: *USD 100*

Input: *end*

500.00 JPY is 145.00 THB

5.00 USD is 159.95 THB

**Example 3** (user inputs are in *italics*)

7.00 USD is 223.93 THB

9.00 EUR is 320.58 THB

18.20 USD is 582.22 THB

1234.00 JPY is 357.86 THB

|  |  |  |
| --- | --- | --- |
| Input: | *USD* | *7* |
| Input: | *EUR* | *9* |
| Input: | *USD* | *18.20* |
| Input: | *JPY* | *1234* |
| Input: | *end* |  |

**Example 4** (user inputs are in *italics*)

8.00 USD is 255.92 THB

5000.00 JPY is 1450.00 THB

|  |  |  |
| --- | --- | --- |
| Input: | *USD* | *8* |
| Input: | *ABC* | *3* |
| ERROR |  |  |
| Input: | *JPY* | *5000* |
| Input: | *end* |  |

Write a program to repeatedly take two inputs animal name and amount then keep the values in a list. The program repeats until user inputs “done”. The program will then display the total amount of each type of animals as a summary. (in a format shown in examples below).

**Example 1** (user inputs are in *italics*)

Input: *done* Empty List ###Summary###

**Example 2** (user inputs are in *italics*)

Input: *snail 2*

Input: *ant 2*

Input: *dog 3*

Input: *fish 2*

Input: *dog 1*

Input: *fish 3* Input: *done* ###Summary###

Total Number of snail : 2 Total Number of ant : 2 Total Number of dog : 4 Total Number of fish : 5

Write a coin deposit program to repeatedly take user input then store the values inside list. The program repeats until user inputs “done. The program will then display the total amount of each type of coin type user deposit as well as the total deposit amount. The program prints "Invalid Input" If the input is not numeric.

**Example 1** (user inputs are in *italics*) Welcome to coin deposit machine Please insert coin: *done*

<-----Deposit Summary >

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 1 | baht | coins | -> | 0 |
| 2 | baht | coins | -> | 0 |
| 5 | baht | coins | -> | 0 |

10 baht coins -> 0 Deposit Amount: 0 baht

**Example 2** (user inputs are in *italics*) Welcome to coin deposit machine Please insert coin: *10*

You inserted 10 baht coin Current Balance: 10 baht Please insert coin: *20*

Only 1,2,5 and 10 baht coin are allowed Please insert coin: *done*

<-----Deposit Summary >

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 1 | baht | coins | -> | 0 |
| 2 | baht | coins | -> | 0 |
| 5 | baht | coins | -> | 0 |

10 baht coins -> 1 Deposit Amount: 10 baht