|  |
| --- |
| Question 1: |
|  |

|  |
| --- |
| Write a program that calculates and prints the value according to the given formula: |
|  |

|  |
| --- |
| Q = Square root of [(2 \* C \* D)/H] |
|  |

|  |
| --- |
| Following are the fixed values of C and H: |
|  |

|  |
| --- |
| C is 50. H is 30. |
|  |

|  |
| --- |
| D is the variable whose values should be input to your program in a comma-separated sequence. |
|  |

|  |
| --- |
| Example |
|  |

|  |
| --- |
| Let us assume the following comma separated input sequence is given to the program: |
|  |

|  |
| --- |
| 100,150,180 |
|  |

|  |
| --- |
| The output of the program should be: |
|  |

18,22,24

ANS: import math

def calculate\_value(input\_sequence, C, H):

result = []

for D in input\_sequence:

Q = int(math.sqrt((2 \* C \* int(D)) / H))

result.append(Q)

return result

# Test the program

if \_\_name\_\_ == "\_\_main\_\_":

C = 50

H = 30

input\_sequence = input("Enter comma-separated values for D: ").split(',')

try:

input\_sequence = [int(D) for D in input\_sequence]

output\_sequence = calculate\_value(input\_sequence, C, H)

print("Output:", ", ".join(map(str, output\_sequence)))

except ValueError:

print("Invalid input! Please enter integer values separated by commas.")

Example usage:

Enter comma-separated values for D: 100,150,180

Output: 18, 22, 24

|  |
| --- |
| Question 2: |
| Write a program which takes 2 digits, X,Y as input and generates a 2-dimensional array. The element value in the i-th row and j-th column of the array should be i\*j. | |
|  | |

|  |
| --- |
| Note: i=0,1.., X-1; j=0,1,¡­Y-1. |
|  |

|  |
| --- |
| Example |
|  |

|  |
| --- |
| Suppose the following inputs are given to the program: |
|  |

|  |
| --- |
| 3,5 |
|  |

|  |
| --- |
| Then, the output of the program should be: |
|  |

|  |
| --- |
| [[0, 0, 0, 0, 0], [0, 1, 2, 3, 4], [0, 2, 4, 6, 8]] |
| ANS: You can create a Python program that takes two digits, X and Y, as input and generates a 2-dimensional array based on the given formula i\*j. Here's the program:  def generate\_2d\_array(X, Y):  array\_2d = [[i \* j for j in range(Y)] for i in range(X)]  return array\_2d  # Test the program  if \_\_name\_\_ == "\_\_main\_\_":  try:  X, Y = map(int, input("Enter two digits (X, Y): ").split(','))  if X < 0 or Y < 0:  print("Both X and Y should be non-negative integers.")  else:  result = generate\_2d\_array(X, Y)  print("Output:")  for row in result:  print(row)  except ValueError:  print("Invalid input! Please enter two integers separated by a comma.")  In this program, the `generate\_2d\_array` function takes two integers X and Y as input and creates a 2-dimensional array based on the given formula i\*j. The function uses list comprehensions to construct the array. The outer loop iterates over the range of X, and the inner loop iterates over the range of Y. Each element in the array is calculated as i \* j.  Example usage:  Enter two digits (X, Y): 3,5  Output:  [0, 0, 0, 0, 0]  [0, 1, 2, 3, 4]  [0, 2, 4, 6, 8] |
|  |

Question 3:

|  |
| --- |
| Write a program that accepts a comma separated sequence of words as input and prints the words in a comma-separated sequence after sorting them alphabetically. |
|  |

|  |
| --- |
| Suppose the following input is supplied to the program: |
|  |

|  |
| --- |
| without,hello,bag,world |
|  |

|  |
| --- |
| Then, the output should be: |
|  |

bag,hello,without,world

ANS: def sort\_words(input\_sequence):

words\_list = input\_sequence.split(',')

sorted\_words = sorted(words\_list)

return ','.join(sorted\_words)

# Test the program

if \_\_name\_\_ == "\_\_main\_\_":

input\_sequence = input("Enter comma-separated words: ")

output\_sequence = sort\_words(input\_sequence)

print("Output:", output\_sequence)

EXAMPLE USAGE:

Enter comma-separated words: without,hello,bag,world

Output: bag,hello,without,world

Question 4:

|  |
| --- |
| Write a program that accepts a sequence of whitespace separated words as input and prints the words after removing all duplicate words and sorting them alphanumerically. |
|  |

|  |
| --- |
| Suppose the following input is supplied to the program: |
|  |

|  |
| --- |
| hello world and practice makes perfect and hello world again |
|  |

|  |
| --- |
| Then, the output should be: |
|  |

again and hello makes perfect practice world

ANS: def remove\_duplicates\_and\_sort(input\_sequence):

words\_list = input\_sequence.split()

unique\_words = sorted(set(words\_list))

return ' '.join(unique\_words)

# Test the program

if \_\_name\_\_ == "\_\_main\_\_":

input\_sequence = input("Enter whitespace-separated words: ")

output\_sequence = remove\_duplicates\_and\_sort(input\_sequence)

print("Output:", output\_sequence)

Example usage:

Enter whitespace-separated words: hello world and practice makes perfect and hello world again

Output: again and hello makes perfect practice world

Question 5:

|  |
| --- |
| Write a program that accepts a sentence and calculate the number of letters and digits. |
|  |

|  |
| --- |
| Suppose the following input is supplied to the program: |
|  |

|  |
| --- |
| hello world! 123 |
|  |

|  |
| --- |
| Then, the output should be: |
|  |

|  |
| --- |
| LETTERS 10 |
|  |

DIGITS 3

ANS: def count\_letters\_digits(sentence):

num\_letters = sum(c.isalpha() for c in sentence)

num\_digits = sum(c.isdigit() for c in sentence)

return num\_letters, num\_digits

# Test the program

if \_\_name\_\_ == "\_\_main\_\_":

input\_sentence = input("Enter a sentence: ")

letters, digits = count\_letters\_digits(input\_sentence)

print("LETTERS", letters)

print("DIGITS", digits)

Example usage:

Enter a sentence: hello world! 123

LETTERS 10

DIGITS 3

Question 6:

|  |
| --- |
| A website requires the users to input username and password to register. Write a program to check the validity of password input by users. |
|  |

|  |
| --- |
| Following are the criteria for checking the password: |
|  |

|  |
| --- |
| 1. At least 1 letter between [a-z] |
|  |

|  |
| --- |
| 2. At least 1 number between [0-9] |
|  |

|  |
| --- |
| 1. At least 1 letter between [A-Z] |
|  |

|  |
| --- |
| 3. At least 1 character from [$#@] |
|  |

|  |
| --- |
| 4. Minimum length of transaction password: 6 |
|  |

|  |
| --- |
| 5. Maximum length of transaction password: 12 |
|  |

|  |
| --- |
| Your program should accept a sequence of comma separated passwords and will check them according to the above criteria. Passwords that match the criteria are to be printed, each separated by a comma. |
|  |

|  |
| --- |
| Example |
|  |

|  |
| --- |
| If the following passwords are given as input to the program: |
|  |

|  |
| --- |
| ABd1234@1,a F1#,2w3E\*,2We3345 |
|  |

|  |
| --- |
| Then, the output of the program should be: |
|  |

ABd1234@1

ANS: You can create a Python program that checks the validity of passwords according to the given criteria. Here's the program:

import re

def is\_valid\_password(password):

# Use regular expressions to check the password against the criteria

if (6 <= len(password) <= 12 and

re.search(r"[a-z]", password) and

re.search(r"[A-Z]", password) and

re.search(r"[0-9]", password) and

re.search(r"[$#@]", password)):

return True

else:

return False

# Test the program

if \_\_name\_\_ == "\_\_main\_\_":

input\_passwords = input("Enter comma-separated passwords: ")

passwords\_list = input\_passwords.split(',')

valid\_passwords = [password for password in passwords\_list if is\_valid\_password(password)]

print("Valid passwords:", ", ".join(valid\_passwords))

```

In this program, the `is\_valid\_password` function takes a password as input and checks it against the given criteria using regular expressions. If the password satisfies all the criteria, it returns `True`; otherwise, it returns `False`.

The program prompts the user to enter a sequence of comma-separated passwords, splits them into a list, and then filters the list to find the valid passwords based on the criteria. Finally, it prints the valid passwords.

Example usage:

Enter comma-separated passwords: ABd1234@1,a F1#,2w3E\*,2We3345

Valid passwords: ABd1234@1