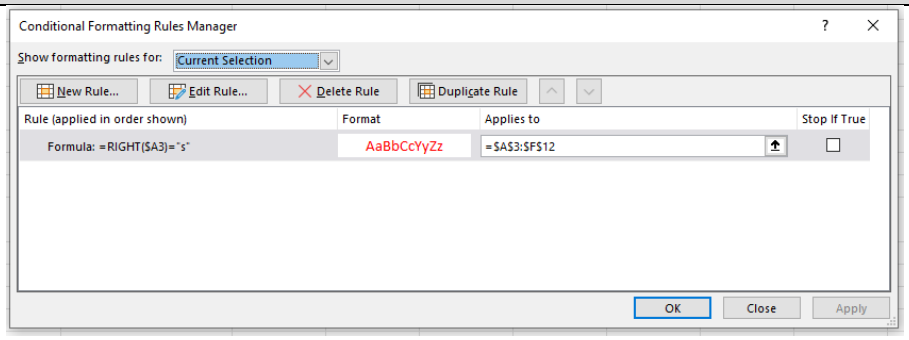
2022 SEC 4 COMPUTING PRELIM PAPER 2 MARKING SCHEME

Task 1

Question	Answer
1	=MEDIAN(B3:B12)
Question	**Answer**
2	=SUMIF(C3:C12,">=10",B3:B12)
Question	**Answer**
3	=B3*C3
Question	**Answer**
4	=SUM(D3:D12)
Question	**Answer**
5	=MATH.FLOOR(D3)
Question	**Answer**
6	=IF(COUNTIF(B$19:F$19,A3)=1,HLOOKUP(A3,B$19:F$20,2,FALSE),"1")

Question	Answer
7	

Task 2

Question	Answer

8	Input of `list_length` from user, including input message.
	
	`list_length = int(input("Please input the length of the list: ")`
	`for i in range(list_length):`

Question	**Answer**
9	Flip the inequality sign in the function. `if arr[p] < arr[p+1]:`
Question	**Answer**

10	Looping the input.
	Checking if input is numeric using `.isdigit()` or equivalent.
	Checking for possibility of negative sign in the first index
	Shifting the typecasting to int from the input line to the line where valid input is added to the list
	Error message if input is not valid
	
	`while True:`
	`    num = input("Please input an integer: ")`
	`    if num.isdigit() or (num[0] == "-" and num[1:].isdigit()):`
	`        arr = arr + [int(num)]`
	`        break`
	`    else:`
	`        print("Invalid input!")`

Question	**Answer**

11	Looping through the list
	Comparing the numbers in the list
	Removing duplicate numbers OR Adding non-duplicate numbers to a second list
	-1m if all this is not done in UDF
	
	`#after the list has been sorted in the UDF`
	`arr2 = [arr[0]]`
	`i = 1`
	`while i < len(arr):`
	`    if arr[i] != arr[i-1]:`
	`        arr2 = arr2 + [arr[i]]`
	`    i += 1`

Sample Code:

```
list_length = int(input("Please input the length of the list: "))
def bubblesort(arr):
    for r in range(1, list_length):
        for p in range(0, list_length - r):
            if arr[p] < arr[p+1]: #flip the inequality sign
                arr[p], arr[p+1] = arr[p+1], arr[p]

    arr2 = [arr[0]]
    i = 1
    while i < len(arr):
        if arr[i] != arr[i-1]:
            arr2 = arr2 + [arr[i]]
        i += 1

    return arr2

arr = []
for i in range(list_length):
    while True:
        num = input("Please input an integer: ")
        if num.isdigit() or (num[0] == "-" and num[1:].isdigit()):
            arr = arr + [int(num)]
            break
        else:
            print("Invalid input!")

print("The sorted list:", bubblesort(arr))
```

Task 3

Question	Answer

12	print('Welcome to Buttercup Customised Bouquets!')
	print('Enter up to three flowers to create your bouquet.')
	
	flowers = ['red roses', 'pink roses', 'buttercups', "baby's breath", 'limonium']
	#syntax: inverted commas
	prices = [7, 9, 10, 3, 5]
	
	counter = 0
	cost = 5 #logic: including a base cost of 5
	while counter < 3: #logic: while instead of if
	    flower = input('Please choose a flower: ')
	    flower = flower.lower() #logic: lower instead of islower
	    if flower not in flowers:
	        print('Sorry, we do not have that flower. Please try again!')
	        continue #syntax: missing indent
	    flower_index = flowers.index(flower)
	    cost += prices[flower_index] #logic: flower_index instead of flower_index+1
	    counter += 1
	    if counter < 3: #logic: 3 instead of 2
	        cont = input('Please enter "N" to end, or any other key to continue: ').upper()
	        if cont == 'N': break #syntax: double == sign; logic: 'N' instead of 'n'
	
	if counter == 3:
	    cost = round(0.9 * cost, 2) #syntax: round function instead of int
	
	print('The cost of your bouquet is ${}.'.format(cost))

Task 4

Question	Answer
13	Proper naming of method signature and variable
	Use math.ceil or round to derived cost
	Return appropriate variable
Question	**Answer**
14	Input statement for N, user_weight
	Input statement for X and S with .split()
	Input validation for X and S
	Calling get_cost() to calculate cost
	Display all N companies name and cost quotes
	All input and output must have appropriate prompts.
Question	**Answer**
15	Output matches stored program
	Results are accurate, as reflected below

	`Enter the number of companies: 5`
	`Enter 5 number of weight (X): 20 30 12 40 50`
	`Enter 5 number of cost (S): 10 17 5.5 18 25.5`
	
	`Enter the estimated weight to be moved: 144`
	`Weight to move: 144`
	`Company Cost`
	`A 80.0`
	`B 85.0`
	`C 66.0`
	`D 72.0`
	`E 76.5`
	`>>>`

Question	**Answer**
16	Using loop to find lowest (a) price quote based on `user_weight`
	Display company(s) with the cheapest price quote
	Calculate lowest (b) cost per kilogram of `user_weight`
	Display company(s) with the cheapest price quote per kilogram
Question	**Answer**
17	Test case 1: TW = 18, MW = 6
	Test case 2: TW = 20, MW = 5
	Test case 3: TW = 30, MW = 8

Sample Code:

```
# Q13
import math

def get_quotes(weight, price, est_weight):
    weight_req = math.ceil(est_weight/weight)
    return price * weight_req

#non math module solution
def get_quotes2(weight, price, est_weight):
    weight_req = int((est_weight/weight)+0.5)
    return price * weight_req

#driver test
print(get_quotes(20, 10, 30))
print(get_quotes2(20, 10, 30))

# Q14
# continue from Q13
N = int(input("Enter the number of companies: "))
validated = False
while not validated:
    x = input("Enter {} number of weights (X): ".format(N)).split()
    if len(x) != N:
        print("Invalid number of weights (X) entered. Please re-
enter.")
    else:
        validated = True
        for i in range(N):
            x[i] = float(x[i])

validated = False
while not validated:
    s = input("Enter {} number of costs (S): ".format(N)).split()
    if len(s) != N:
        print("Invalid number of costs (S) entered. Please re-enter.")
    else:
        validated = True
        for i in range(N):
            s[i] = float(s[i])
print()
user_weight = int(input("Enter the estimated weight to be moved: "))

costs = []
print("Weight to move: {}".format(user_weight))
print("Company\tQuotes")
for i in range(N):
    temp = get_quotes(x[i], s[i], user_weight)
    costs.append(temp)
    print("{}\t{}".format(chr(65+i), costs[i]))
```

```
# Q15
Enter the number of companies: 5
Enter 5 number of weights (X): 20 35 40 50
Invalid number of weights (X) entered. Please re-enter.
Enter 5 number of weights (X): 20 35 12 40 50
Enter 5 number of costs (S): 10 17 5.5 18 23 25.5
Invalid number of costs (S) entered. Please re-enter.
Enter 5 number of costs (S): 10 17 5.5 18 25.5

Enter the estimated weight to be moved: 144
Weight to move: 144
Company Quotes
A       80.0
B       85.0
C       66.0
D       72.0
E       76.5
>>>
```

```
# Q16
# continue from Q14
coy_per_userwt, coy_per_kilo = 'A', 'A'
cost_per_userwt, cost_per_kilo = costs[0], (s[0]/x[0])
for i in range(N):
    if cost_per_userwt > costs[i]:
        cost_per_userwt = costs[i]
        coy_per_userwt = chr(65+i)
    temp = s[i] / x[i]
    if temp < cost_per_kilo:
        cost_per_kilo = temp
        coy_per_kilo = chr(65+i)

print()
print("Cheapest quote based on user weight: {} at
${}".format(coy_per_userwt,cost_per_userwt))
print("Cheapest quote based on per kilo: {} at
${}".format(coy_per_kilo,cost_per_kilo))

# Q17
def maxCombo(tw, mw):
    if (tw < mw):
        return 0
    count = 1
    for i in range(mw, int(tw/2)+1):
        count += maxCombo(tw-i, i)

    return count



#Driver Test
print("TW:{} MW:{} -> {}".format(18, 6, maxCombo(18, 6)))
print("TW:{} MW:{} -> {}".format(20, 5, maxCombo(20, 5)))
print("TW:{} MW:{} -> {}".format(30, 8, maxCombo(30, 8)))
```