Marking scheme (4E Computing Prelim)

Task 1

1	=ROUND(AVERAGE(C4:E4), 0)	2	1m for AVERAGE( ) 1m for ROUND( )
2	==MAX(F4:F12)-MIN(F4:F12)	1	
3	=IF(OR(F4=MAX($F$4:$F$12), F4=LARGE($F$4:$F$12, 2)), "Favourite", "") OR =IF(F4=MAX($F$4:$F$12), "Favourite", IF(F4=LARGE($F$4:$F$12, 2), "Favourite", "")	3	1m for MAX( ) or LARGE( ) 1m for OR( ) 1m IF( ) 1m for MAX( ) or LARGE( ) 1m for IF( ) 1m for nested IF()
4	=(HLOOKUP(F4, $B$17:$E$19, 3, TRUE)+100)/100*B4 OR =(HLOOKUP(F4, $B$17:$E$19, 3, TRUE)/100 + 1)*B4 OR = HLOOKUP(F4, $B$17:$E$19, 3, TRUE)/100*B4 + B4	2	1m for HLOOK( ) 1m for correct calculation
5	Correct conditional formatting	2	

Task 2

6

```
message = input("Enter a message: ")

encrypted_m = ""

for i in range(len(message)):

    if message[i] in "aeiou":

        encrypted_m = encrypted_m + message[i].upper()

    elif message[i] == " ":

        encrypted_m = encrypted_m + "&"

    else:

        encrypted_m = encrypted_m + chr(ord(message[i])+3)

print("Encrypted message:", encrypted_m)
```

1m – check for vowel

1m – convert to upper case

1m – check for space

1m – convert to "&" and

include else: for the rest of the cases

7

```
def decrypt(s):
    original_m = ""
    for i in range(len(s)):
        if s[i] in "AEIOU":
            original_m = original_m + s[i].lower()
        elif s[i] == "&":
            original_m = original_m + " "
        else:
            original_m = original_m + chr(ord(s[i])- 3)
    return original_m


# main program
print("Original message:", decrypt(encrypted_m))
```

1m – initialise local variable

1m – check for vowel and convert to lower case

1m – check for "&" and convert to space

1m – convert to 3 characters in front

1m – return decrypted message

1m – call function correctly and print

Task 3

8

```
def correct_range(num, limit):
    return 0 <= num <= limit

print("Time in 24-hour format are entered as 4 digits")
valid = False
while valid == False:
    time = input("Enter a time in 24-hour format: ")
    if len(time) != 4:
        print("Length of input must be 4")
    elif time.isdigit() == False:
        print("Input must be digits only")
    else:
        hour_int = int(time[:2])
        minute_str = time[2:]
        if not correct_range(hour_int, 23):
            print("Hour must be in the range 0 to 23 inclusive")
        elif not correct_range(int(minute_str), 59):
            print("Minute must be in the range 0 to 59 inclusive")
        else:
            valid = True
```

```
if correct_range(hour_int, 23) == False:
```

```
if correct_range(int(minute_str), 59) == False:
```

```
if 0 <= hour_int <= 11:
    indicator = "AM"
else:
    indicator = "PM"

new_hour = hour_int
if hour_int == 0:
    new_hour = hour_int + 12
elif hour_int >= 13:
    new_hour = hour_int - 12

converted_time = str(new_hour) + ":" + minute_str + indicator
print(converted_time)
```

Task 4

9	initialisation of 2 lists	1m	
	while loop	1m	FOR loop not acceptable
	3 suitable input messages	1m	
	appropriate error message	1m	
	input name/mark and store in list	1m	either name or mark
	while loop to validate marks correctly and re-prompt	1m	
	update counter and total	1m	
	update variable for loop continuation	1m	
	correct output (count and average)	1m	with appropriate message
	Max	9 m	

Accept if students use sum(), len()

9	initialisation of 2 lists	1m	
	while loop	1m	FOR loop not acceptable
	3 suitable input messages	1m	
	appropriate error message	1m	
	input name/mark and store in list	1m	either name or mark
	while loop to validate marks correctly and re-prompt	1m	
	use of sum() and len() to find average	1m	
	update variable for loop continuation	1m	
	correct output (count and average)	1m	with appropriate message
	Max	9 m	

10	Enter student name: Student A Enter student mark: -1 Marks must be between 0 to 100 inclusive Re-enter student mark: 0 Any more students? Enter "y" or "n": y Enter student name: Student B Enter student mark: 101 Marks must be between 0 to 100 inclusive Re-enter student mark: 100 Any more students? Enter "y" or "n": y Enter student name: Student C Enter student mark: 40 Any more students? Enter "y" or "n": y Enter student name: Student D Enter student mark: 78 Any more students? Enter "y" or "n": y Enter student name: Student E Enter student mark: 34 Any more students? Enter "y" or "n": y Enter student name: Student F Enter student mark: 51 Any more students? Enter "y" or "n": y Enter student name: Student G Enter student mark: 63 Any more students? Enter "y" or "n": n There are 7 students The average mark is 52.3	Test data – 2 marks: 1m – two types of test data 2m – three types of test data Output – 1 mark: 1m – correct output based on test data

11	loop management for checking	1m	
	correct condition	1m	
	store in list	1m	
	correct output for 0 remedial	1m	must not be followed by number of tutors
	appropriate output message	1m	
	correct output for remedial	1m	
	correct calculation for number of tutor	1m	
	correct output for tutor	1m	with appropriate message
	Max	8 m	

Sample program

Q8

```
name_list = []
mark_list = []
count = 0
total = 0

more_student = "y"
while more_student == "y":
    name = input("Enter student name: ")
    name_list = name_list + [name]

    mark = int(input("Enter student mark: "))
    while mark < 0 or mark > 100:
        print("Marks must be between 0 to 100 inclusive")
        mark = int(input("Re-enter student mark: "))
    mark_list = mark_list + [mark]
    total = total + mark
    count = count + 1
    more_student = input('Any more students? Enter "y" or "n": ').lower()

average = total / count
print("There are", count, "students")
print("The average mark is", round(average, 1))
```

Sample program

Q10

```
remedial_list = []
for i in range(len(mark_list)):
    if mark_list[i] < average:
        remedial_list = remedial_list + [name_list[i]]

num_remedial = len(remedial_list)
if num_remedial == 0:
    print("There are no students for remedial")
else:
    print("Students identified for remedial class")
    for j in range(num_remedial):
        print(remedial_list[j])

    num_tutor = num_remedial // 3
    if num_remedial % 3 != 0:
        num_tutor = num_tutor + 1
    print("Number of peer tutors needed:", num_tutor)
```