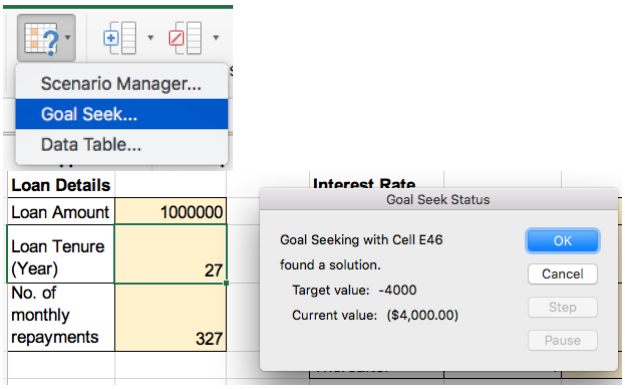
2018 SEC 4 COMPUTING PRELIM PAPER 2 MARKING SCHEME

Task 1

Question	Answer	Marks
1	One mark for the correct formula. ==B3*12 or =B3*12 or =B$3*12 or =$B3*12	**1**
2	One mark for the correct values. **Interest Rate** Year 1 1.70% Year 2 1.70% Year 3 1.70% Thereafter 4 1.70%	**1**
3(a)	One mark for the correct formula, with B4 as a fixed reference. =IF(ROW(B10)-9>B4, "", ROW(B10)-9) (Or equivalent)	**1**
3(b)	One mark for the correct formula.. =IF(B10="","",CEILING(B10/12,1)) =IF(B11="","",CEILING(B11/12,1)) … =IF(B489="","",CEILING(B489/12,1)) (Or equivalent)	**1**
3(c)	One mark for the correct formula. =IF(B10="","",ABS(PMT(C10/12,B4,B2))) (Or equivalent)	**1**
3(d)	One mark for the correct formula. =IF(B10="", "", ABS(IPMT(C10/12, B10, B4, B2))) (Or equivalent)	**1**
3(e)	One mark for each correct formulae. In cell G10: =IF(B10="", "", E10-F10) In cell H10: =IF(B10="", "", D10-G10) (Or equivalent)	**2**

3(f)	One mark for copying formulae A10:H10 to rows 11 to 489.	**1**
4	One mark for the correct length of tenure.  He should set the loan tenure to **27 years and 3 months**.	**1**

Task 2

Question	Answer	Marks
5(a)	`size = 10`	**1**
5(b)	One mark for the correct validation criterion One mark for printing feedback AND asking for input again. Insert between line 4 and 5:	**2**

```
while True:
  try:
    income =  int(input("({})Annual income in $: ".format(employee+1)))
    if income < 0 or income > 120000:
       raise Exception
    except:
        print("Please enter a value from 0 to 120000!")
    else:
        break

OR

  income =  int(input("({})Annual income in $: ".format(employee+1)))
  while income < 0 or income > 120000:
    income = print("Please enter a value from 0 to 120000!")
```

5(c)	One mark for the correct code to obtain the highest tax. One mark for printing the value.	**2**

<table>
<tr>
<td></td>
<td>

```
highestTax = 0
```

<table>
<tr>
<td>

```
if tax > highestTax:
        highestTax = tax
        n = employee + 1
```

</td>
<td>OR</td>
<td>

```
highestTax =
max(highestTax, tax)
```

</td>
</tr>
</table>

```
print("Highest tax payable is $", round(highestTax,2))
```

</td>
<td></td>
</tr>
<tr>
<td>5(d)</td>
<td>One mark for the printing the correct employee.

```
if tax > highestTax:
        highestTax = tax
        n = employee + 1

print("Employee {} paid the highest tax.".format(n))
```

</td>
<td>1</td>
</tr>
<tr>
<td>5(e)</td>
<td>One mark for the correct code to calculate the required percentage.
One mark for printing the value.

```
count = 0
…
if income <= 20000:
        tax = 0
        count += 1
…
print("Percentage who do not need to pay tax:
{}%.".format(round(count/size*100,1)))
```

</td>
<td>2</td>
</tr>
<tr>
<td>6</td>
<td>Correct the four highlighted parts:

```
    elif income <= 30000:
        tax = (income-20000) * 0.02
    elif income <= 40000:
        tax = 200 + (income-30000) * 0.035
    elif income <= 80000:
        tax = 550 + (income-40000) * 0.07
    else:
        tax = 2800 + (income-80000) * 0.115
```

</td>
<td>2</td>
</tr>
</table>

Task 3

Question	Answer	Marks
7		**10**

```
s = 0
count = 0

while True:
  x = input('Enter a positive integer. Type "done" to finish.')
  if x == "done":
    break
  elif not x.isdigit():
    print ("Invalid input. Try again.")
  else:
    x = int(x)
    if count == 0:
      M = m = x
    else:
      M = max(M, x)
      m = min(m, x)
    s += x
    count += 1

if count==0:
  average = s = M = m = "NA"

average = round(s/count, 1)
print("\nYou have entered {} number(s).".format(count))
print("The sum of the number entered is {}.".format(s))
print("\nThe average of the number entered is {}.".format(average))
print("\nThe maximum of the number entered is {}.".format(M))
print("The minimum of the number entered is {}.".format(m))
```

Task 4

Question	Answer	Marks
8		**12**

```
while True:
    i = input('Enter a string of digits or space:')
    if any([not x in '0123456789 ' for x in i]):
        print('Input error! Try again!')
    else:
        break

F = [i.count(x) for x in '0123456789 ']
B = [x for x in i.split() if sum([int(y) for y in x])>=20]

blocks = 0 if i.isspace() else 1
s = 0
count = 0
for x in range(1, len(i)-1):
    if i[x]==' ' and i[x+1]!=' ':
        blocks += 1

for x in '0123456789':
    print('Frequency of {}: {}'.format(x, F[int(x)]))

print('\nNumber of block(s): {}'.format(blocks))

print('Block(s) with sum 20 or more:')
for i, b in enumerate(B):
    print('({}) {}'.format(i+1, b))
```

1 mark of asking for user input.
1 mark for correctly validating if input is either a digit or space.
1 mark for feedback and re-requesting input if entered input is invalid.
1 mark for tracking the frequencies of each digit, and
1 mark for storing the frequencies
1 mark for initialising a variable to track the number of blocks
2 marks for the correct code to count the number of blocks
1 mark for the loop to print each frequency
2 marks for printing the right frequencies. (-1 mark for each mistake)
1 mark to print the number of blocks

Question	Answer	Marks
9	1 mark of to enter the correct input 1 mark for saving the png file. 1 mark with the correct name. Enter : 3647 94859 8482 3209 832 45346 Frequency of 0: 1 Frequency of 1: 0 Frequency of 2: 3 Frequency of 3: 4 Frequency of 4: 5 Frequency of 5: 2 Frequency of 6: 2 Frequency of 7: 1 Frequency of 8: 4 Frequency of 9: 3 Number of block(s): 6	**3**

10	1 mark to split input in blocks.
	1 mark to sum the digits in each block.
	1 mark to check if the sum obtained is 20 or more.
	1 mark to print block numbers.
	1 mark to print correct blocks.
	
	`while True:`
	`    i = input('Enter a string of digits or space:')`
	`    if any([not x in '0123456789 ' for x in i]):`
	`        print('Input error! Try again!')`
	`    else:`
	`        break`
	
	`F = [i.count(x) for x in '0123456789 ']`
	`B = [x for x in i.split() if sum([int(y) for y in x])>=20]`
	
	`blocks = 0 if i.isspace() else 1`
	`s = 0`
	`count = 0`
	`for x in range(1, len(i)-1):`
	`    if i[x]==' ' and i[x+1]!=' ':`
	`        blocks += 1`
	
	`for x in '0123456789':`
	`    print('Frequency of {}: {}'.format(x, F[int(x)]))`
	
	`print('\nNumber of block(s): {}'.format(blocks))`
	
	`print('Block(s) with sum 20 or more:')`
	`for i, b in enumerate(B):`
	`    print('({}) {}'.format(i+1, b))`
		5