

## Programming Languages Homework 2

功課習題：

1. 3.17
2. 3.18
3. 3.19
4. 3.20
5. 3.33

6. 4.15
7. 4.16
8. 4.27
9. 4.28
10. 4.31

繳交期限：10/20 (四) 晚上 11:59 前

繳交格式：103360001\_李奇樺.pdf

繳交內容：讀書會(包含:組員、討論時間、地點、照片或 Google Meet)、心得報告(包含: 心得、GitHub 程式連結、GitHub 的截圖)

上傳位置：Homework\Upload

帳號、密碼：CC

如有問題請 Mail 至

陳彥霖 david87124@gmail.com

張浩維 howard3317@gmail.com

莊家郡 kk69347321@gmail.com

林承孝 maxjg5019@gmail.com

## 3.17 :

**3.17 (Credit Limit Calculator)** Develop a C program that will determine if a department store customer has exceeded the credit limit on a charge account. For each customer, the following facts are available:

- a) Account number
- b) Balance at the beginning of the month
- c) Total of all items charged by this customer this month
- d) Total of all credits applied to this customer's account this month
- e) Allowed credit limit

The program should input each fact, calculate the new balance ( $= \text{beginning balance} + \text{charges} - \text{credits}$ ), and determine whether the new balance exceeds the customer's credit limit. For those customers whose credit limit is exceeded, the program should display the customer's account number, credit limit, new balance and the message "Credit limit exceeded." Here is a sample input/output dialog:

```
Enter account number (-1 to end): 100
Enter beginning balance: 5394.78
Enter total charges: 1000.00
Enter total credits: 500.00
Enter credit limit: 5500.00
Account: 100
Credit limit: 5500.00
Balance: 5894.78
Credit Limit Exceeded.

Enter account number (-1 to end): 200
Enter beginning balance: 1000.00
Enter total charges: 123.45
Enter total credits: 321.00
Enter credit limit: 1500.00

Enter account number (-1 to end): 300
Enter beginning balance: 500.00
Enter total charges: 274.73
Enter total credits: 100.00
Enter credit limit: 800.00

Enter account number (-1 to end): -1
```

## 3.18 :

**3.18 (Sales Commission Calculator)** One large chemical company pays its salespeople on a commission basis. The salespeople receive \$200 per week plus 9% of their gross sales for that week. For example, a salesperson who sells \$5000 worth of chemicals in a week receives \$200 plus 9% of \$5000, or a total of \$650. Develop a program that will input each salesperson's gross sales for last week and will calculate and display that salesperson's earnings. Process one salesperson's figures at a time. Here is a sample input/output dialog:

```
Enter sales in dollars (-1 to end): 5000.00
Salary is: $650.00

Enter sales in dollars (-1 to end): 1234.56
Salary is: $311.11

Enter sales in dollars (-1 to end): -1
```

### 3.19 至 3.20 :

**3.19 (Interest Calculator)** The simple interest on a loan is calculated by the formula  

$$\text{interest} = \text{principal} * \text{rate} * \text{days} / 365;$$

The preceding formula assumes that rate is the annual interest rate, and therefore includes the division by 365 (days). Develop a program that will input principal, rate and days for several loans, and will calculate and display the simple interest for each loan, using the preceding formula. Here is a sample input/output dialog:

```
Enter loan principal (-1 to end): 1000.00
Enter interest rate: .1
Enter term of the loan in days: 365
The interest charge is $100.00

Enter loan principal (-1 to end): 1000.00
Enter interest rate: .08375
Enter term of the loan in days: 224
The interest charge is $51.40

Enter loan principal (-1 to end): -1
```

**3.20 (Salary Calculator)** Develop a program that will determine the gross pay for each of several employees. The company pays "straight time" for the first 40 hours worked by each employee and pays "time-and-a-half" for all hours worked in excess of 40 hours. You're given a list of the employees of the company, the number of hours each employee worked last week and the hourly rate of each employee. Your program should input this information for each employee and should determine and display the employee's gross pay. Here is a sample input/output dialog:

```
Enter # of hours worked (-1 to end): 39
Enter hourly rate of the worker ($00.00): 10.00
Salary is $390.00

Enter # of hours worked (-1 to end): 40
Enter hourly rate of the worker ($00.00): 10.00
Salary is $400.00

Enter # of hours worked (-1 to end): 41
Enter hourly rate of the worker ($00.00): 10.00
Salary is $415.00
```

### 3.33 :

**3.33 (Hollow Rectangle of Plus Symbols)** Modify the program you wrote in Exercise 3.32 so that it prints a hollow rectangle. For example, if your program reads a length of 3 and a breadth of 12, it should print

```
+++++++
+       +
+++++++
```

**4.16 (Triangle-Printing Program)** Write a program that prints the following patterns separately, one below the other. Use `for` loops to generate the patterns. All asterisks (\*) should be printed by a single `printf` statement of the form `printf("%5s", " ");` (this causes the asterisks to print side by side). [Hint: The last two patterns require that each line begin with an appropriate number of blanks.]

(A)	(B)	(C)	(D)
0	0000000000	0000000000	0
1	0000000001	0000000001	1
2	0000000010	0000000010	2
3	0000000011	0000000011	3
4	0000000100	0000000100	4
5	0000000101	0000000101	5
6	0000000110	0000000110	6
7	0000000111	0000000111	7
8	0000001000	0000001000	8
9	0000001001	0000001001	9
10	0000001010	0000001010	10
11	0000001011	0000001011	11
12	0000001100	0000001100	12
13	0000001101	0000001101	13
14	0000001110	0000001110	14
15	0000001111	0000001111	15
16	0000010000	0000010000	16
17	0000010001	0000010001	17
18	0000010010	0000010010	18
19	0000010011	0000010011	19
20	0000010100	0000010100	20
21	0000010101	0000010101	21
22	0000010110	0000010110	22
23	0000010111	0000010111	23
24	0000011000	0000011000	24
25	0000011001	0000011001	25
26	0000011010	0000011010	26
27	0000011011	0000011011	27
28	0000011100	0000011100	28
29	0000011101	0000011101	29
30	0000011110	0000011110	30
31	0000011111	0000011111	31
32	0000100000	0000100000	32
33	0000100001	0000100001	33
34	0000100010	0000100010	34
35	0000100011	0000100011	35
36	0000100100	0000100100	36
37	0000100101	0000100101	37
38	0000100110	0000100110	38
39	0000100111	0000100111	39
40	0000101000	0000101000	40
41	0000101001	0000101001	41
42	0000101010	0000101010	42
43	0000101011	0000101011	43
44	0000101100	0000101100	44
45	0000101101	0000101101	45
46	0000101110	0000101110	46
47	0000101111	0000101111	47
48	0000110000	0000110000	48
49	0000110001	0000110001	49
50	0000110010	0000110010	50
51	0000110011	0000110011	51
52	0000110100	0000110100	52
53	0000110101	0000110101	53
54	0000110110	0000110110	54
55	0000110111	0000110111	55
56	0000111000	0000111000	56
57	0000111001	0000111001	57
58	0000111010	0000111010	58
59	0000111011	0000111011	59
60	0000111100	0000111100	60
61	0000111101	0000111101	61
62	0000111110	0000111110	62
63	0000111111	0000111111	63
64	0001000000	0001000000	64
65	0001000001	0001000001	65
66	0001000010	0001000010	66
67	0001000011	0001000011	67
68	0001000100	0001000100	68
69	0001000101	0001000101	69
70	0001000110	0001000110	70
71	0001000111	0001000111	71
72	0001001000	0001001000	72
73	0001001001	0001001001	73

4.27、28：

**4.28 (Calculating Weekly Pay)** A company pays its employees as managers (who receive a fixed weekly salary), hourly workers (who receive a fixed hourly wage for up to the first 40 hours they work and “time-and-a-half”—i.e., 1.5 times their hourly wage—for overtime hours worked), commission workers (who receive \$250 plus 5.7% of their gross weekly sales), or pieceworkers (who receive a fixed amount of money for each of the items they produce—each pieceworker in this company works on only one type of item). Write a program to compute the weekly pay for each employee. You do not know the number of employees in advance. Each type of employee has its own pay code: Managers have paycode 1, hourly workers have code 2, commission workers have code 3 and pieceworkers have code 4. Use a `switch` to compute each employee’s pay based on that employee’s paycode. Within the `switch`, prompt the user (i.e., the payroll clerk) to enter the appropriate facts your program needs to calculate each employee’s pay based on that employee’s paycode. [Note: You can input values of type `double` using the conversion specifier `%lf` with `scanf`.]

4.31 :

**4.31** (*Diamond-Printing Program*) Write a program that prints the following diamond shape. You may use `printf` statements that print either a single asterisk (\*) or a single blank. Maximize your use of repetition (with nested `for` statements) and minimize the number of `printf` statements.

```
  *
 ***
*****
*****
*****
*****
*****
 ***
  *
```