OOP 期末上機考

B 卷

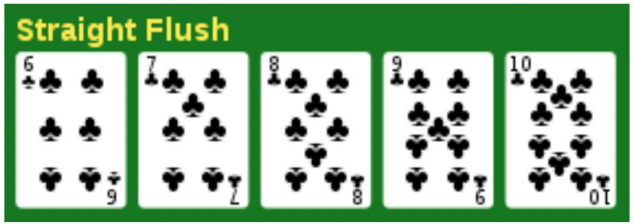
1.STL vector - **POKER(20%)**

Description

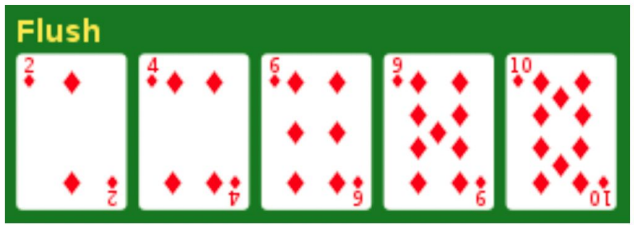
In poker, players construct hands of five cards according to predetermined rules, which vary according to which variant of poker is being played. A hand always consists of five cards. The suits of the cards are used to determine whether a hand forms a flush or straight flush.

In this case, there can be several categories of poker hands as follow pages.

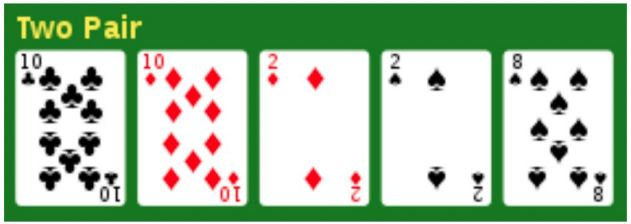
Straight flush



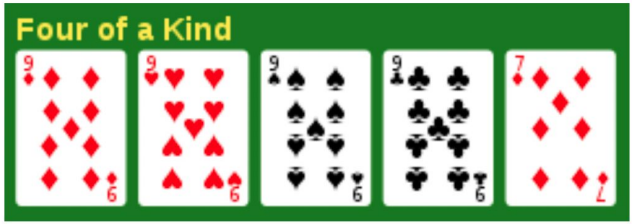
Flush



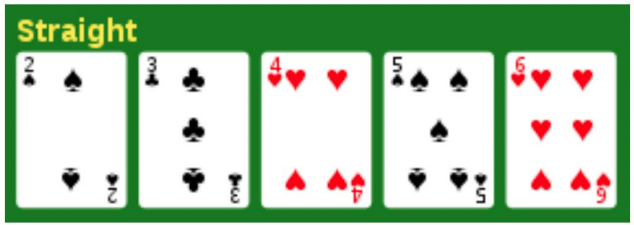
Two pair



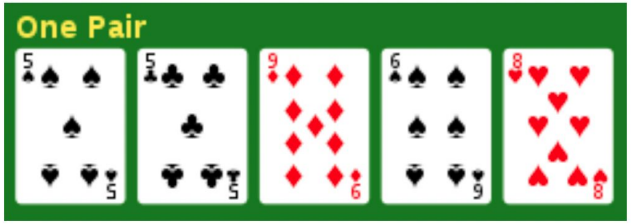
Four of a kind



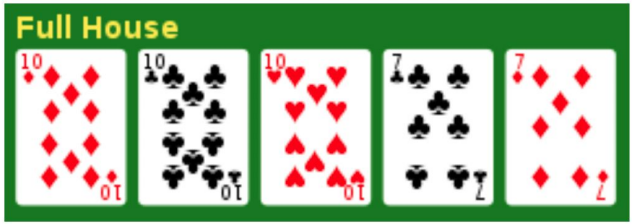
Straight



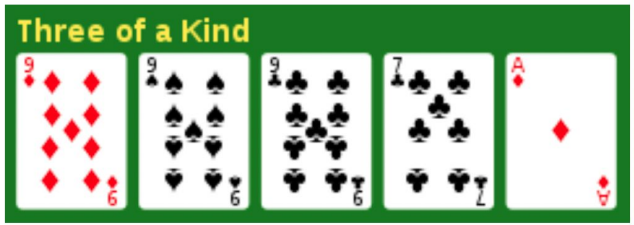
One pair



Full house



Three of a kind



Input

- The first line is the number of test cases. Each line of the test cases will contain five different strings separated by a single space to represent five cards. Each string consists two characters. The first one is **the number** of the card and the second one is **the suit** of the card.

- The card numbers are A,2,3,4,5,6,7,8,9,X,J,Q,K.Note that the numbers more than 9 are represented as X, J, Q, K, and 1 is represented as A. And the suits are S, H, D, C, all in capital letter, represent spade, heart, diamond and club. For your convenience, the five cards will be listed in **descending sequence.**

- Input **ends** with a single row with the integer 0.

Output

- Input testcase: **input_pokerB.txt**
- Print the category of the poker hands and **a single empty row** between each group of test cases.

Sample Input	**Sample Output**
5 XC 9C 8C 7C 6C 9D 9H 9S 9C 7D XD XC XH 7C 7D XD 9D 6D 4D 2D 6H 5S 4H 3C 2S 3 9D 9S 9C 7C AD XC XD 8S 2D 2S 9D 8H 6S 5S 5C 0	Straight flush Four of a kind Full house Flush Straight Three of a kind Two pair One pair

2-STL Map**(20%)**

Given an array of unsigned integers and two other unsigned integers.

Example:

Input:

array = 2, 8, 56, 1, 0

x = 6

y = 4

Output:2

Input:

array = 0, 1, 2

x = 1

y = 1

output: 1

Description:

You have to convert the integers in the array into binary numbers. Find the size of the minimum subset of the array that there are at least 6 0's and 4 1's .

2 = 10, 8 = 1000, 56 =111000, 1 = 1, 0 = 0 , The subsets include {"0", "1", "10", "1000", "111000"}, {"0", "1", "10", "1000"}, {"0", "1", "10", "111000"}, {"0",

"1", "1000", "111000"}, {"0", "10", "1000", "111000"}, {"1", "10", "1000", "111000"}... {"1000", "111000"}, .The minimum size of the subset is 2.

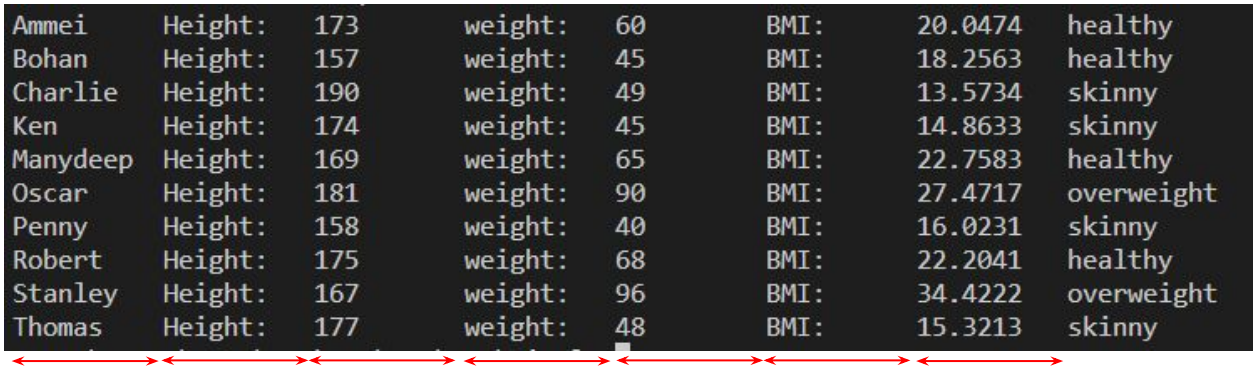
3-Classes(25%)

Define and implement a new class “student_data”

- These are 3 private data members
  - name
  - height
  - weight

- Record all student’s data and calculate their BMI and physical conditions. BMI = kilograms/(meters^2).
- Record height, weight and count BMI with data type double

- Output: cout left and setw(10) ⟷
  - student's information
    - Student's data
    - BMI
    - overweight (BMI>27) , skinny(BMI<17) or healthy
  - Sort student's name ascending
  - Output example:



```
Ammei     Height:   173       weight:   60        BMI:      20.0474   healthy
Bohan     Height:   157       weight:   45        BMI:      18.2563   healthy
Charlie   Height:   190       weight:   49        BMI:      13.5734   skinny
Ken       Height:   174       weight:   45        BMI:      14.8633   skinny
Manydeep  Height:   169       weight:   65        BMI:      22.7583   healthy
Oscar     Height:   181       weight:   90        BMI:      27.4717   overweight
Penny     Height:   158       weight:   40        BMI:      16.0231   skinny
Robert    Height:   175       weight:   68        BMI:      22.2041   healthy
Stanley   Height:   167       weight:   96        BMI:      34.4222   overweight
Thomas    Height:   177       weight:   48        BMI:      15.3213   skinny
```

4-Inheritance(25%)

Assume that we have a brand-new calculating method for the salary of the NBA rookies, given the list of the draft result (Input.txt), output the draft pick、the name and the salary of the players in the output.txt.

With Input.txt, build up the following base and derived classes
Base class : player, with private data member (1) name (2) salary (3) pick

Class name	Derived from	Additional private data member
Second_round_pick (31~60)	player	Base_salary
First_round_pick (1~30)	player	Base_salary
Lottery_pick (1~14)	First_round_pick	Lottery_ magnification
Top_5_pick (1~5)	Lottery_pick	Top_5_Bonus

4-Inheritance

- Base salary for 1~30 picks is 4M, and 1.5M for 31~60 picks.
- For 1~30 picks, 0.15M more salary for every going up of the pick.
- For 31~60 picks, 0.06M more salary for every going up of the pick.
- For the number one overall pick, his lottery magnification is 20%, and every one pick lower, 1% less of the lottery magnification (1~14 picks have lottery magnification).
- For the number one overall pick, his top-5 bonus is 5M, and every one pick lower, 1M less of the bonus (1~5 picks have top 5 bonus).
- Output format (all the information is left-aligned) : pick → setw(3) / name → setw(23)
- Final salary calculation :

  Original salary * (1+lottery magnification) + top-5 bonus

pick	salary
1	8.35 x (1+ 0.2) + 5 = 15.02
6	7.6 x (1 + 0.15) = 8.74
15	6.25

1 Cade Cunningham	31 Isaiah Todd
2 Jalen Green	32 Jeremiah Robinson-Earl
3 Evan Mobley	33 Jason Preston
4 Scottie Barnes	34 Rokas Jokubaitis
5 Jalen Suggs	35 Herbert Jones
6 Josh Giddey	36 Miles McBride
7 Jonathan Kuminga	37 JT Thor
8 Franz Wagner	38 Ayo Dosunmu
9 Davion Mitchell	39 Neemias Queta
10 Ziaire Williams	40 Jared Butler
11 James Bouknight	41 Joe Wieskamp
12 Joshua Primo	42 Isaiah Livers
13 Chris Duarte	43 Greg Brown
14 Moses Moody	44 Kessler Edwards
15 Corey Kispert	45 Juhann Begarin
16 Alperen Sengun	46 Dalano Banton
17 Trey Murphy III	47 David Johnson
18 Tre Mann	48 Sharife Cooper
19 Kai Jones	49 Marcus Zegarowski
20 Jalen Johnson	50 Filip Petrusev
21 Keon Johnson	51 Brandon Boston Jr.
22 Isaiah Jackson	52 Luka Garza
23 Usman Garuba	53 Charles Bassey
24 Josh Christopher	54 Sandro Mamukelashvili
25 Quentin Grimes	55 Aaron Wiggins
26 Nah'Shon Hyland	56 Scottie Lewis
27 Cameron Thomas	57 Balsa Koprivica
28 Jaden Springer	58 Jericho Sims
29 Day'Ron Sharpe	59 RaiQuan Gray
30 Santi Aldama	60 Georgios Kalaitzakis

1	Cade Cunningham	15.02	31	Isaiah Todd	3.24
2	Jalen Green	13.758	32	Jeremiah Robinson-Earl	3.18
3	Evan Mobley	12.499	33	Jason Preston	3.12
4	Scottie Barnes	11.243	34	Rokas Jokubaitis	3.06
5	Jalen Suggs	9.99	35	Herbert Jones	3
6	Josh Giddey	8.74	36	Miles McBride	2.94
7	Jonathan Kuminga	8.493	37	JT Thor	2.88
8	Franz Wagner	8.249	38	Ayo Dosunmu	2.82
9	Davion Mitchell	8.008	39	Neemias Queta	2.76
10	Ziaire Williams	7.77	40	Jared Butler	2.7
11	James Bouknight	7.535	41	Joe Wieskamp	2.64
12	Joshua Primo	7.303	42	Isaiah Livers	2.58
13	Chris Duarte	7.074	43	Greg Brown	2.52
14	Moses Moody	6.848	44	Kessler Edwards	2.46
15	Corey Kispert	6.25	45	Juhann Begarin	2.4
16	Alperen Sengun	6.1	46	Dalano Banton	2.34
17	Trey Murphy	5.95	47	David Johnson	2.28
18	Tre Mann	5.8	48	Sharife Cooper	2.22
19	Kai Jones	5.65	49	Marcus Zegarowski	2.16
20	Jalen Johnson	5.5	50	Filip Petrusev	2.1
21	Keon Johnson	5.35	51	Brandon Boston	2.04
22	Isaiah Jackson	5.2	52	Luka Garza	1.98
23	Usman Garuba	5.05	53	Charles Bassey	1.92
24	Josh Christopher	4.9	54	Sandro Mamukelashvili	1.86
25	Quentin Grimes	4.75	55	Aaron Wiggins	1.8
26	Nah'Shon Hyland	4.6	56	Scottie Lewis	1.74
27	Cameron Thomas	4.45	57	Balsa Koprivica	1.68
28	Jaden Springer	4.3	58	Jericho Sims	1.62
29	Day'Ron Sharpe	4.15	59	RaiQuan Gray	1.56
30	Santi Aldama	4	60	Georgios Kalaitzakis	1.5

5-Templates**(30%)**

Write a template for a function called countItemFrequency that accepts as parameters a vector and a value which may be contained in the vector. Iterate through the vector and count the number of occurrences of the value in the vector and return the count to the user. with a custom class with the == operator overloaded.

You have to input 24(int) and A(char), and output the number of occurrences of the value in the vector.

```
16
14
A
6
15
A
20
16
C
24
13
15
C
19
D
24
15
24
B
24
15
23
15
6
12
8
17
19
B
33
15
14
D
15
8
B
12
18
C
15
```

(10%)

input:24

output:4(the number of occurrences of the value)

(10%)

input:B

output:3(the number of occurrences of the value)