OOP&DS Final (2018/6/11)

Q1(20%)

Submit Q1.h

Your boss want you to extend more function on stl vector function which type is integer. So you must create your own class myVector that inherit Vector<int> and extend 4 functions:

(1) sum : retrun sum of the whole vector which type is integer

(2) concat : concat another integer vector

(3) sort : sort the whole vector

(4) show : show the whole vector value

However, you're in the bad mood. You want to play a trick. You overwrite a size function that always return zero:

(5) size : return zero always

Example:

```
1  myVector myvector1,myvector2;
2  cout<< myvector1.size() <<endl;
3  for (int i=0; i<5; i++) myvector1.push_back(i);
4  for (int i=10; i>5; i--) myvector2.push_back(i);
5  cout<< myvector1.size() <<endl;
6  cout<< "sum :" << myvector1.sum() <<endl;
7  myvector1.concat(myvector2);
8  cout<< "sum :" << myvector1.sum() <<endl;
9  myvector1.show();
10 myvector1.sort();
11 myvector1.show();
```

Output:

0

0

sum :10

sum :50

0 1 2 3 4 10 9 8 7 6

0 1 2 3 4 6 7 8 9 10

Implement your own .h Class file

Q2(20%)

Submit Q2.h

Please complete the cpp file. In addition, you are not allowed to modify header file and make share you code can work with the following main.cpp (Do not write any main function in your cpp)

```cpp
#include<iostream>
#include<cstdio>
#include<cstdlib>
#include<cstring>
#include<cmath>
#include "ZipCode.h"

using namespace std;

int main()
{
    ZipCode zc1;
    cout<<zc1<<endl;              //???
    ZipCode zc2(zc1);
    zc1.extend("AB");
    zc2.extend(98);
    cout<<zc1<<endl<<zc2<<endl;  //???-AB     ???-98

    ZipCode zc3(123);
    zc3.extend("CD");
    cout<<zc3<<endl;     //123-CD
    ZipCode zc4(zc3);
    cout<<zc4<<endl;  //123-CD

    ZipCode zc5("END");
    cout<<zc5<<endl;  //END
    return 0;
}
```

(1)You need to complete constructor and destructor：ZipCode()、ZipCode(int)、ZipCode(char*)、ZipCode(ZipCode&)、~ZipCode()

- ZipCode()：Initial the code to “???”
- ZipCode(123)：Initial the code to “123”
- ZipCode(ABC)：Initial the code to “ABC”
- ZipCode(ZipCode&)：Copy the code

  for example：

  ZipCode zc1;

  ZipCode zc2(zc1);

  then copy zc1 to zc2。

Be careful, after copy constructor, zc1 and zc2 are independent, which means if you modify zc1 after copy, zc2 won’t be modified.

(2)You need to complete the following function and operator overloading：extend(int)、extend(char*)、ostream& operator << (ostream&out, const ZipCode&)

- extend(int)、extend(char*)：Extend the code to the format “XXX-XX”

  for example：

  Zipcode zc1(“ABC”);

  zc1.extend(45);

  Zipcode zc2(“123”);

zc2.extend(CD);

then zc1：ABC-45 and zc2：123-CD

- ostream& operator << (ostream&out, const ZipCode&)：print the code (Only print code! Don't print any newline and space in this function!)

Q3(20%)

Submit Q3.cpp (including main function)

Please implement **quick sort** and print the result of each step.

Note:

1. choose **the first** element as pivot in each step.
2. the length of the unsort list <= 1000
3. each number <= 10 ^ 9

input:

[the number of unsort list]
[the unsort number list1]
[the unsort number list2]
...

output:

[the number list after the first sorting]
[the number list after the second sorting]
…
[the final result of the sorted number list]

example:

Your program is to read from standard input.

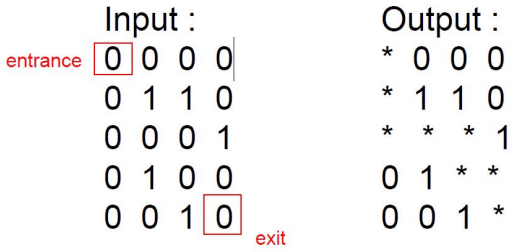
input	2 // two unsorted list 9 8 2 1 6 // the first list 26 5 37 1 61 11 59 15 48 19 // the second list
output	6 8 2 1 9 // the first result 2 1 6 8 9 1 2 6 8 9 1 2 6 8 9 11 5 19 1 15 26 59 61 48 37 // the second result 1 5 11 19 15 26 59 61 48 37 1 5 11 19 15 26 59 61 48 37 1 5 11 15 19 26 59 61 48 37 1 5 11 15 19 26 48 37 59 61 1 5 11 15 19 26 37 48 59 61 1 5 11 15 19 26 37 48 59 61

*seperated by space

Q4(20%)

Submit Q4.cpp (including main function)

We'll give you a MxN maze map(M >= 1, N >= 1), and the position of entrance and exit are in (1,1) and (m,n). Number 0 represent the road and number 1 represent the wall. For example, below is a 5x4 maze,the entrance is as (1,1) and the exit is at (5,4). Please find the shortest path and label the path by *.



If there's no way to reach the exit, just output "NOWAY".

You should solve this problem with stack. Other methods such as recursion are forbidden. Make sure that you find the shortest path. All of the mazes will only have one path to reach the exit.

example:
Your program is to read from standard input.

Input	2 // num of maze 4 3 // MxN maze 0 1 0 0 0 1 0 0 0 0 1 0 2 5 // MxN maze 0 1 0 1 0 0 0 0 1 0
Output	* 1 0 // answer of the first maze * 0 1 * * * 0 1 * NOWAY //answer of the second maze

Hint:
1.We provide complete "ArrayStack.h". You can use either "ArrayStack.h" or stack library to solve the problem.
2.

```
typedef struct {
  int x;
  int y;
  int dir;
} item;
item stack[m*p];
```

Q5(20%)

Submit Q5.cpp (including main function)

Please output the (1)preorder traversal, (2)postorder traversal, and (3)level-order traversal of the given binary search tree.

Input

Your program is to read from standard input.

The input consists of T test cases. The number of test cases T is given in the first line.

Each test case starts with a line containing an integer N representing the number of keys, 1 ≤ N ≤ 20. In the next line, N numbers are given. Insert the numbers to the binary search tree one by one. There is a single space between the integers.

Output

For each test case, the output must follow the description below. The outputs of two consecutive cases will be separated by a **blank line**.

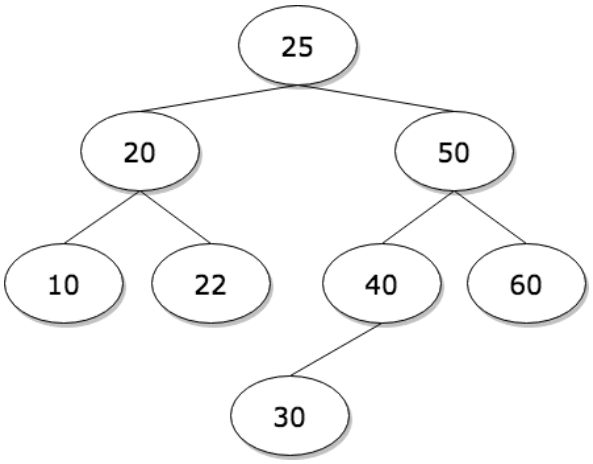
For each case, the output contains three lines:

They are respectively **preorder, postorder, level-order traversal** of the input binary tree.

example:

input	2 8 25 20 10 22 50 40 60 30 4 10 20 30 40
output	25 20 10 22 50 40 30 60 10 22 20 30 40 60 50 25 25 20 50 10 22 40 60 30 10 20 30 40 40 30 20 10

Example BST:



Submission:

1. Submit your file to e3, and name it with “Q[question number].cpp or.h”. EX: Q1.h, Q5.cpp.
2. Submit only one cpp or header file for each question.
3. Upload the corresponding file of each question to e3.
4. Do not compress it.
5. Make sure your code can run on NCTU CS workstation successfully.