1057. Stack (30)

时间限制

150 ms

内存限制

65536 kB

代码长度限制

16000 B

判题程序

Standard

作者

CHEN, Yue

Stack is one of the most fundamental data structures, which is based on the principle of Last In First Out (LIFO). The basic operations include Push (inserting an element onto the top position) and Pop (deleting the top element). Now you are supposed to implement a stack with an extra operation: PeekMedian -- return the median value of all the elements in the stack. With N elements, the median value is defined to be the (N/2)-th smallest element if N is even, or ((N+1)/2)-th if N is odd.

**Input Specification:**

Each input file contains one test case. For each case, the first line contains a positive integer N (<= 105). Then N lines follow, each contains a command in one of the following 3 formats:

Push *key*  
Pop  
PeekMedian

where *key* is a positive integer no more than 105.

**Output Specification:**

For each Push command, insert *key* into the stack and output nothing. For each Pop or PeekMedian command, print in a line the corresponding returned value. If the command is invalid, print "Invalid" instead.

**Sample Input:**

17

Pop

PeekMedian

Push 3

PeekMedian

Push 2

PeekMedian

Push 1

PeekMedian

Pop

Pop

Push 5

Push 4

PeekMedian

Pop

Pop

Pop

Pop

**Sample Output:**

Invalid

Invalid

3

2

2

1

2

4

4

5

3

Invalid

[提交代码](https://www.patest.cn/contests/pat-a-practise/1057)

自己的代码，时间上很存在问题，这个题目对效率的要求比较严格，所以必须要想出比较高效的算法才能最终运行通过。

/\* pat test exercise \*/

/\* Stack patID:1057 Alex Sun 2016-5-2 Ahu BoNan\*/

/\* URL:https://www.patest.cn/contests/pat-a-practise/1057 \*/

/\* 2016-5-2 partly right. reason:not quickly \*/

#include<iostream>

#include<queue>

#include<string>

#include<stack>

#include<vector>

#include<algorithm>

using namespace std;

class Stack{

public:

vector<int> data;

vector<int> assist;

void push(int t){

data.push\_back(t);

assist.push\_back(t);

}

void pop(){

data.pop\_back();

assist.pop\_back();

}

int top(){

return data.back();

}

bool empty(){

if(data.empty()) return true;

else return false;

}

int size(){

return data.size();

}

};

int main()

{

int commandLines;

vector<string> commands;

vector<int> imiStack;

Stack sta;

int n;

int i;

string command;

cin>>commandLines;

getline(cin,command);

while(commandLines--){

getline(cin,command);

commands.push\_back(command);

}

vector<string>::iterator iter=commands.begin();

for(;iter!=commands.end();iter++){

command=\*iter;

if(command=="Pop"){

if(sta.empty()==true) cout<<"Invalid"<<endl;

else{

cout<<sta.top()<<endl;

sta.pop();

}

}else if(command=="PeekMedian"){

if(sta.empty()==true) cout<<"Invalid"<<endl;

else{

if(sta.size()%2==0) n=sta.size()/2;

else n=(sta.size()+1)/2;

n--;

/\* the follwing using priority\_queue,but not as fast as satisfied \*/

// priority\_queue<int,vector<int>,greater<int> > tempQ(sta.data.begin(),sta.data.end());

// while(i<n){

// tempQ.pop();

// i++;

// }

// cout<<tempQ.top()<<endl;

sort(sta.assist.begin(),sta.assist.end());

cout<<sta.assist.at(n)<<endl;

}

}else{

int num=0;

int cnt\_int=1; //jinwei flag

int pointer=command.length()-1;

while(command[pointer]!=' '){

// cout<<"command[pointer] "<<command[pointer]<<endl;

num+=cnt\_int\*(command[pointer]-'0'); //attention: can not cast directely

cnt\_int\*=10;

pointer--;

}

sta.push(num);

}

}

return 0;

}

有几个测试用例没有通过，下面来看一看别人做的比较高效的算法。

#include<iostream>

#include<algorithm>

#include<cstdio>

#include<vector>

#include<map>

#include<set>

#include<queue>

#include<stack>

#include<functional>

#include<string>

#include<iomanip>

using namespace std;

vector<int> coll;

vector<int> tmp;

priority\_queue<int> pq;

int main()

{

#ifdef ONLINE\_JUDGE

#else

freopen("D:\\in.txt", "r", stdin);

freopen("D:\\out.txt", "w", stdout);

#endif

int n(0);

scanf("%d", &n);

string str;

char ch[20];

int t(0);

for (int i = 0; i < n; i++)

{

scanf("%s", ch);

if (ch[1] == 'o')

{

if (coll.empty())

{

printf("%s\n", "Invalid");

continue;

}

else

{

printf("%d\n", coll[coll.size() - 1]);

coll.pop\_back();

}

}

else if (ch[1] == 'u')

{

scanf("%d", &t);

coll.push\_back(t);

}

else if (ch[1] == 'e')

{

while (!pq.empty())

pq.pop();

if (coll.empty())

{

printf("%s\n", "Invalid");

continue;

}

else if (coll.size()==1)

printf("%d\n", coll[0]);

else if (coll.size()%2!=0)

{

int num = coll.size();

int t = (num + 1) / 2;

for (int k = 0; k < num; k++)

{

if (pq.size() < t)

{

pq.push(coll[k]);

}

else

{

if (coll[k] < pq.top())

{

pq.pop();

pq.push(coll[k]);

}

}

}

printf("%d\n", pq.top());

}

else

{

while (!pq.empty())

pq.pop();

int num = coll.size();

int t = num/ 2;

for (int k = 0; k < num; k++)

{

if (pq.size() <t)

{

pq.push(coll[k]);

}

else

{

if (coll[k] < pq.top())

{

pq.pop();

pq.push(coll[k]);

}

}

}

printf("%d\n", pq.top());

}

}

}

return 0;

}