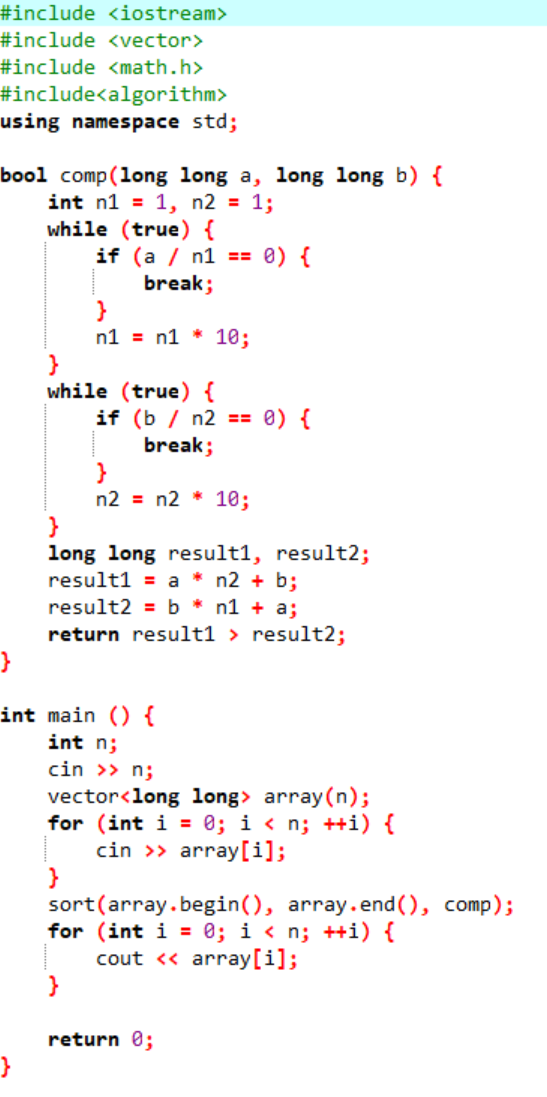
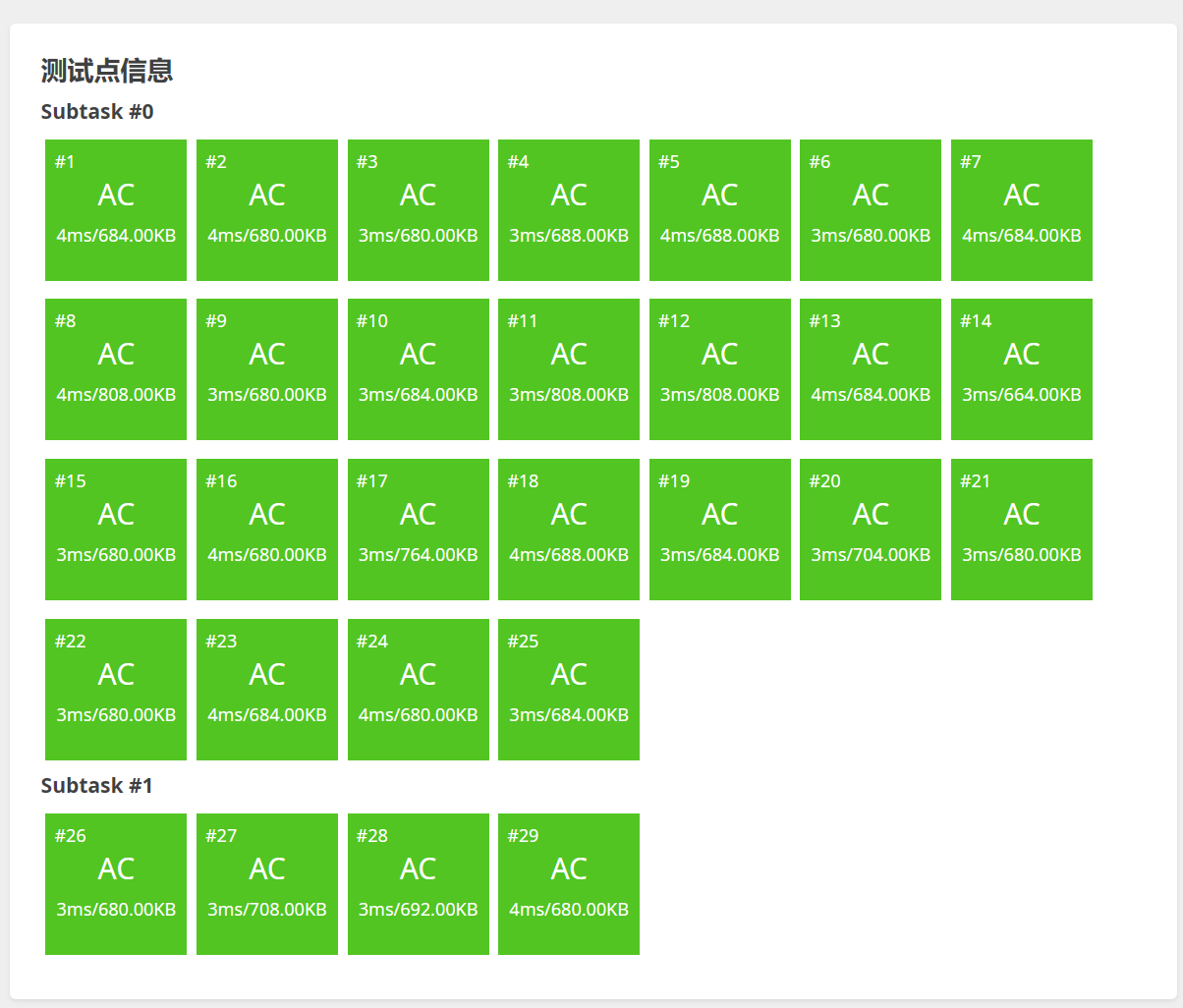
1. 最大多位整数问题





1. 单源最短路径

#include <iostream>

using namespace std;

struct conn{

int node;

int value;

conn\* next;

};

int main() {

int n, m, s;

conn\*\* lines;

int\* nowdistance;

int\* result;

cin >> n;

cin >> m;

cin >> s;

lines = (conn\*\*)malloc(n \* sizeof(conn\*));

if (!lines) {

exit(-1);

}

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

lines[i] = (conn\*)malloc(sizeof(conn));

if (!lines[i]) {

exit(-1);

}

lines[i]->node = i;

lines[i]->value = 0;

lines[i]->next = NULL;

}

for (int i = 0; i < m; ++i) {

int node1;

int node2;

int value;

cin >> node1;

cin >> node2;

cin >> value;

--node1;

--node2;

conn \*newp;

newp = (conn\*)malloc(sizeof(conn));

if(!newp) {

exit(-1);

}

newp->node = node2;

newp->value = value;

newp->next = lines[node1]->next;

lines[node1]->next = newp;

}

result = (int\*)malloc(n \* sizeof(int));

if (!result) {

exit(-1);

}

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

result[i] = -1;

}

result[s - 1] = 0;

nowdistance = (int\*)malloc(n \* sizeof(int));

if (!nowdistance) {

exit(-1);

}

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

nowdistance[i] = -1;

}

for (conn\* p = lines[s - 1]; p != NULL; p = p->next) {

if (nowdistance[p->node] == -1) {

nowdistance[p->node] = p->value;

}

else {

if (nowdistance[p->node] > p->value) {

nowdistance[p->node] = p->value;

}

}

}

for (int i = 0; i < n - 1; ++i) {

int min = 2147483647;

int rec\_j;

int flag = 0;

for (int j = 0; j < n; ++j) {

if (nowdistance[j] != 0 && nowdistance[j] != -1) {

if (min > nowdistance[j]) {

min = nowdistance[j];

rec\_j = j;

}

flag = 1;

}

}

if (flag == 0) {

break;

}

result[rec\_j] = min;

nowdistance[rec\_j] = 0;

for (conn\* p = lines[rec\_j]; p != NULL; p = p->next) {

if (nowdistance[p->node] != 0) {

int tem = p->value + result[rec\_j];

if (nowdistance[p->node] > tem || nowdistance[p->node] == -1) {

nowdistance[p->node] = tem;

}

}

}

}

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

if (result[i] == -1) {

cout << 2147483647 << " ";

}

else {

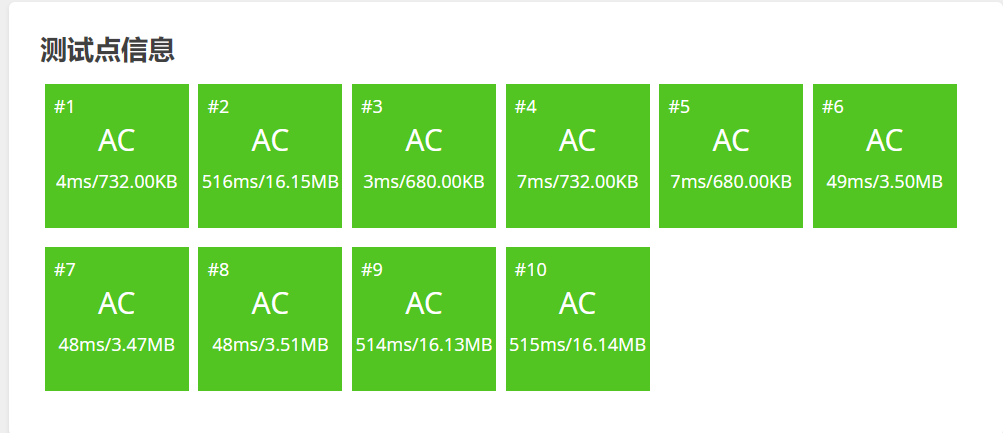
cout << result[i] << " ";

}

}

return 0;

}



1. 摆动序列



