

上机实验8 2023080050 林瞳  
计37

代码	2
测试	18
测试 1 - 无LOG, 无BOX	18
测试 2 - 有LOG, 无BOX	18
测试 3 - 无LOG, 有BOX	20
测试 4 - 有LOG, 有BOX	21

# 代码

C/C++

```
// Author: 2023080050 林瞳 Pau Tong Lin Xu
// 计37, 程序设计基础 1-2
// Date: 2023/12/10
// The program below generates a matrix and performs matrix
operations.
// Configure the initial matrix generation in 'config.txt'
// Type a list of commands in 'cmd.txt'
// Output in terminal and 'log.txt' (if specified)

#include <iostream>
#include <iomanip>
#include <fstream>
#include <string>
#include <vector>
#include <cstdlib>
#include <ctime>
#include <cmath>

using namespace std;

// SIZE is determined at runtime
// so I use a 2d pointer to allocate dynamic memory to my matrix
int SIZE;
int** matrix;

string LOG;
ofstream fout;

bool isLog = false;
bool BOX = false;
```

```

int getRandom(int max){ // returns an integer between 1 and max.
    // The generation seed is created in the main() function.
    return (rand()*max/RAND_MAX)+1;
}

void output(string text, bool endl, int width){ // outputs in
terminal (and log file)
    // the function will output in log file if global 'isLog'
variable is true
    // for formatting purposes, the function also takes in the width
and line skips as parameters

    if(width != 0){
        cout << setw(width);
        fout << setw(width);
    }
    if (isLog) {
        cout << text;
        fout << text;
        if (endl){
            cout << endl;
            fout << endl;
        }
    }
    else{
        cout << text;
        if (endl){
            cout << endl;
        }
    }
}

```

```

void normalOutput(){ // outputs the matrix in non-box format
    for (int i = 0; i<SIZE ; i++){
        output("----", false, 0);
    }
    output("", true, 0);
    for (int r = 0 ; r<SIZE ; r++){
        for (int c=0; c<SIZE ; c++){
            if (matrix[r][c] == -1){
                output("*", false, 3);
            }
            else{
                output(to_string(matrix[r][c]), false, 3);
            }
            output(" ", false, 0);
        }
        output("", true, 0);
    }
    for (int i = 0; i<SIZE ; i++){
        output("----", false, 0);
    }
    output("", true, 0);
}

int getMax(){ // gets the maximum length of a values in the matrix
    int max = 0;
    for (int i = 0 ; i<SIZE ; i++){
        for (int j = 0 ; j<SIZE ; j++){
            if (to_string(matrix[i][j]).length() > max){
                max = to_string(matrix[i][j]).length();
            }
        }
    }
}

```

```

    }
    return max;
}

void boxedOutput(){ // outputs the matrix in box format
    int width = getMax();

    for (int r = 0 ; r < SIZE ; r++){
        for (int c = 0; c < SIZE ; c++){
            output("+", false, 0);
            for (int i = 0 ; i < width ; i++){
                output("-", false, 0);
            }
        }
        output("+", true, 0);

        for (int c = 0; c < SIZE ; c++){
            output("|", false, 0);
            if (matrix[r][c] == -1){
                output("*", false, width);
            }
            else{
                output(to_string(matrix[r][c]), false, width);
            }
        }
        output("|", true, 0);
    }

    for (int c = 0; c < SIZE ; c++){
        output("+", false, 0);
        for (int i = 0 ; i < width ; i++){
            output("-", false, 0);

```

```

    }

    }

    output("+", true, 0);
}

void outputstep(){ // function chooses whether to use box format
depending on global 'BOX' variable
    if(BOX){
        boxedOutput();
    }
    else{
        normalOutput();
    }
}

void compress(char direction){ // compresses the matrix
    // the direction of compression is passed in a the parameter
    // 'l' = left, 'r' = right, 'u' = up, 'd' = down
    int a, b, r, c;
    for (a = 0; a<SIZE ; a++){
        int sum = 0;
        for (b = 0; b<SIZE ; b++){
            if (direction=='l' || direction=='r'){r = a; c = b;}
            else if (direction=='u' || direction=='d'){c = a; r =
b;}

            sum += matrix[r][c];
            matrix[r][c] = 0;
        }
        if (direction=='l'){matrix[r][0] = sum;}
        else if (direction=='r'){matrix[r][SIZE-1] = sum;}
        else if (direction=='u'){matrix[0][c] = sum;}
        else if (direction=='d'){matrix[SIZE-1][c] = sum;}
    }
}

```

```

    }
}

void rotate(){ // rotates matrix 90 degrees clockwise with respect
to bottom right element
    int** temp;
    temp = new int*[SIZE];
    for (int i = 0; i<SIZE ; i++){
        temp[i] = new int[SIZE];

    }

    for (int r = 0; r < SIZE; r++){
        for (int c = 0; c < SIZE; c++){
            temp[r][c] = matrix[SIZE-1-c][r];
        }
    }

    for (int r = 0; r < SIZE; r++){
        for (int c = 0; c < SIZE; c++){
            matrix[r][c] = temp[r][c];
        }
    }

    for (int i = 0; i<SIZE ; i++){
        delete[] temp[i];
    }
    delete[] temp;
}

void adjust_column(char c){ // moves zeros to the top of column c
    // the order of the remaining elements are maintained
    vector<int> nums(0);
    for (int r=0; r<SIZE; r++){

```

```

        if (matrix[r][c]!=0){
            nums.push_back(matrix[r][c]);
        }
    }
    for (int i=0; i<(SIZE-nums.size()); i++){
        matrix[i][c] = 0;
    }
    int a = 0;
    for (int i=(SIZE-nums.size()); i<SIZE; i++){
        matrix[i][c] = nums[a];
        a++;
    }
}

void ablation(){
    // 1. orders each column using adjust_column()
    // 2. adds pairs
    // 3. orders each column again using adjust_column()
    // 4. adds a new 2 or 4 to a zero element
    for (int c = 0; c<SIZE ;c++){
        adjust_column(c);
        for (int i=0; i<SIZE-1; i++){
            if (matrix[i][c]==matrix[i+1][c]){
                matrix[i][c] = 0;
                matrix[i+1][c]*=2;
                i++;
            }
        }
        adjust_column(c);
    }
    bool exists_zero = false;

```





```

        count += 1;
    }
}

}

}

matrix[r][c] = count;
}

}

}

string remove_space(string line){ // removes spaces from string
    string result = "";
    for (int i = 0; i<line.length(); i++){
        if (line[i] != ' ') {result += line[i];}
    }
    return result;
}

int str_to_num(string str){ // changes a number from string to int
format
    int num = 0;
    int multiple = 1;
    for (int i = str.length()-1; i>=0 ; i--){
        num += (int(str[i])-48)*multiple;
        multiple *= 10;
    }
    return num;
}

bool is_Empty(ifstream& pFile) // checks if a file is empty
{

```

```

        return pFile.peek() == ifstream::traits_type::eof();
    }

bool is_Valid(string num){ // checks if string can be converted to
int
    if (!(num[0]=='-' || (num[0]>='0' && num[0]<='9'))){
        return false;
    }
    for (int i=1; i<num.length() ; i++){
        if (!(num[i]>='0' && num[i]<='9')){
            return false;
        }
    }
    return true;
}

using namespace std;

int main(){

    SIZE = 0;
    int RAND = 0;
    vector<string> commands(0);

    // HANDLING config.txt

    ifstream fin("config.txt");
    if (fin){
        while (!fin.eof()){
            string line;
            getline(fin, line);
            line = remove_space(line);

```

```

        if (line.substr(0,4) == "SIZE"){
            if (is_Valid(line.substr(5, line.length()))){
                SIZE = str_to_num(line.substr(5,
line.length()));
            }
        }
        if (line.substr(0,4) == "RAND"){
            if (is_Valid(line.substr(5, line.length()))){
                RAND = str_to_num(line.substr(5,
line.length()));
            }
        }
        if (line.substr(0,3) == "LOG"){
            LOG = line.substr(4, line.length());
            isLog = true;
        }
        if (line.substr(0,3) == "BOX"){
            if (line.substr(4, line.length()) == "TRUE"){
                BOX = true;
            }
        }
        // any line starting with '#' is disregarded
    }
    if (!(SIZE > 0)){
        cout << "Incorrect 'config.txt' format" << endl;
        SIZE = 4;
    }
    if (!(RAND > 0)){
        RAND = 0;
    }
    fin.close();

```

```

    }
    else{
        SIZE = 4;
        RAND = 0;
        cout << "Incorrect 'config.txt' format: file not found" <<
endl;
    }

    // setting the log file to "LOG"

    if (!LOG.empty()){
        fout.open(LOG);
    }

    // output chosen configuration

    output("SIZE=" + to_string(SIZE) + ", RAND=" + to_string(RAND),
true, 0);

    // HANDLING cmd.txt

    bool cmd_exists = true;
    fin.open("cmd.txt");
    bool isEmpty_cmd = is_Empty(fin);

    if (fin){
        while (!fin.eof()){
            string line;
            getline(fin, line);
            commands.push_back(remove_space(line));
        }
        fin.close();
    }

```

```

    }
    else{
        cout << "'cmd.txt' not found" << endl;
        cmd_exists = false;
    }

    // GENERATING MATRIX

    srand((unsigned int)time(NULL)); // generation seed

    matrix = new int*[SIZE];
    for (int i = 0; i<SIZE; i++){
        matrix[i] = new int[SIZE];
    }

    for (int r = 0 ; r<SIZE ; r++){ // filling the initial matrix
        for (int c=0; c<SIZE ; c++){
            matrix[r][c] = 0;
        }
    }

    int num;
    if (!(RAND > 0)){
        num = 1;
    }
    else {
        num = SIZE*SIZE*RAND/100;
    }

    int count = 0;
    while (count < num){ // randomly selecting elements to be filled
        int max_power = 4;

```

```

        if (num == 1){
            max_power = 2;
        }
        int r = getRandom(SIZE)-1;
        int c = getRandom(SIZE)-1;
        if (matrix[r][c] == 0){
            matrix[r][c] = pow(2, getRandom(max_power));
            count++;
        }
    }

    // MATRIX OPERATIONS

    //ofstream fout("")

    if (!cmd_exists){
        for (int r = 0 ; r<SIZE ; r++){
            for (int c=0; c<SIZE ; c++){
                if (matrix[r][c] != 0){
                    matrix[r][c] = -1;
                }
            }
        }
        asterix_count();
        output("FINAL MATRIX:", true, 0);
        outputstep();
    }
    else if (isEmpty_cmd && cmd_exists){
        for (int r = 0 ; r<SIZE ; r++){
            for (int c=0; c<SIZE ; c++){
                if (matrix[r][c] != 0){
                    matrix[r][c] = getRandom(20);
                }
            }
        }
    }
}

```

```

    }

    }

    }
    output("FINAL MATRIX:", true, 0);
    outputstep();
}
else {
    int counter = 1;

    output("INITIAL MATRIX:", true, 0);
    outputstep();

    for (int i = 0 ; i<commands.size() ; i++){
        if (commands[i][0] == 'r'){
            rotate();
            output(to_string(counter) + ". ROTATION:", true, 0);
            outputstep();
            counter++;
        }
        else if (commands[i][0] == 'a'){
            ablation();
            output(to_string(counter) + ". ABLATION:", true, 0);
            outputstep();
            counter++;
        }
        else if (commands[i][0] == 'c'){
            compress(commands[i][1]);
            output(to_string(counter) + ". COMPRESSION with
direction " + commands[i][1] + ":", true, 0);
            outputstep();
            counter++;
        }
    }
}

```



```
        else if (commands[i][0] == 'Q'){
            output(to_string(counter) + ". QUIT:", true, 0);
            break;
        }
    }
}

// END

fout.close();

for (int i = 0; i < SIZE; ++i) {
    delete[] matrix[i];
}
delete[] matrix;
}
```

# 测试

## 测试 1 - 无LOG, 无BOX

The screenshot shows a C++ IDE with three main panels. The left panel displays 'config.txt' with the following content:

```
1 SIZE = 4
2 RAND = 100
3 BOX = FALSE
4
```

The middle panel displays 'project8.cpp' with the following content:

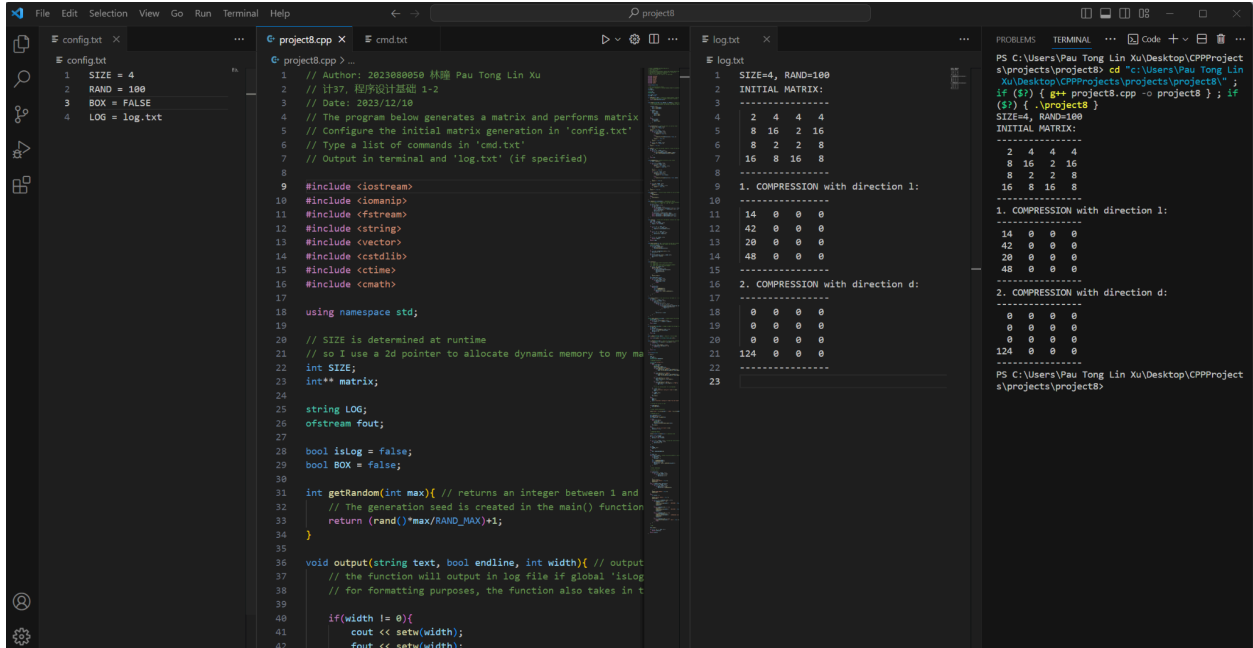
```
1 // Author: 2023080900 林晓 Pau Tong Lin Xu
2 // 计37, 程序设计基础 1-2
3 // Date: 2023/12/10
4 // The program below generates a matrix and performs matrix
5 // Configure the initial matrix generation in 'config.txt'
6 // Type a list of commands in 'cmd.txt'
7 // Output in terminal and 'log.txt' (if specified)
8
9 #include <iostream>
10 #include <iomanip>
11 #include <fstream>
12 #include <string>
13 #include <vector>
14 #include <cstdlib>
15 #include <ctime>
16 #include <cmath>
17
18 using namespace std;
19
20 // SIZE is determined at runtime
21 // so I use a 2d pointer to allocate dynamic memory to my ma
22 int SIZE;
23 int** matrix;
24
25 string LOG;
26 ofstream fout;
27
28 bool isLog = false;
29 bool BOX = false;
30
31 int getRandom(int max){ // returns an integer between 1 and
32 // The generation seed is created in the main() function
33 return (rand()*max/RAND_MAX)+1;
34 }
35
36 void output(string text, bool endl, int width){ // output
37 // the function will output in log file if global 'isLog
38 // for formatting purposes, the function also takes in t
39
40 if(width != 0){
41 cout << setw(width);
42 fout << setw(width);
43 }
```

The right panel displays the terminal output, which shows the execution of the program. The output includes the initial matrix generation and the compression results. The output is as follows:

```
PS C:\Users\Pau Tong Lin Xu\Desktop\CPPProject> cd "C:\Users\Pau Tong Lin Xu\Desktop\CPPProject\projects\project8" && if ($?) { g++ project8.cpp -o project8 }; if ($?) { .\project8 }
SIZE=4, RAND=100
INITIAL MATRIX:
4 2 4 4
4 4 4 2
4 2 4 2
16 4 16 2
-----
1. COMPRESSION with direction 1:
14 0 0 0
14 0 0 0
12 0 0 0
38 0 0 0
-----
2. COMPRESSION with direction d:
0 0 0 0
0 0 0 0
0 0 0 0
78 0 0 0
-----
PS C:\Users\Pau Tong Lin Xu\Desktop\CPPProject> cd "C:\Users\Pau Tong Lin Xu\Desktop\CPPProject\projects\project8" && if ($?) { g++ project8.cpp -o project8 }; if ($?) { .\project8 }
```

如上图所示, 当 'config.txt' 里面没有 'LOG' 且 'BOX' 不等于 'TRUE', 则程序只把结果输出到 terminal 里。该输出不用边框围绕数字。

## 测试 2 - 有LOG, 无BOX



```
1 // Author: 2023080950 林瑞 Pau Tong Lin Xu
2 // 计37 程序设计基础 1-2
3 // Date: 2023/12/10
4 // The program below generates a matrix and performs matrix
5 // Configure the initial matrix generation in 'config.txt'
6 // Type a list of commands in 'cmd.txt'
7 // Output in terminal and 'log.txt' (if specified)
8
9 #include <iostream>
10 #include <iomanip>
11 #include <fstream>
12 #include <string>
13 #include <vector>
14 #include <cstdlib>
15 #include <ctime>
16 #include <cmath>
17
18 using namespace std;
19
20 // SIZE is determined at runtime
21 // so I use a 2d pointer to allocate dynamic memory to my ma
22 int SIZE;
23 int** matrix;
24
25 string LOG;
26 ofstream fout;
27
28 bool isLog = false;
29 bool BOX = false;
30
31 int getRandom(int max){ // returns an integer between 1 and
32 // The generation seed is created in the main() function
33 return (rand()*max/RAND_MAX)+1;
34 }
35
36 void output(string text, bool endl, int width){ // output
37 // the function will output in log file if global 'isLog'
38 // for formatting purposes, the function also takes in t
39
40 if(width != 0){
41 cout << setw(width);
42 fout << setw(width);
43 }
44 if (isLog) {
45 cout << text;
46 fout << text;
47 if (endl){
48 cout << endl;
49 fout << endl;
50 }
51 }
52 else{
53 cout << text;
54 if (endl){
55 cout << endl;
56 }
57 }
58 }
```

```
1 SIZE=4, RAND=100
2 INITIAL MATRIX:
3
4 2 4 4 4
5 8 16 2 16
6 8 2 2 8
7 16 8 16 8
8
9 1. COMPRESSION with direction 1:
10
11 14 0 0 0
12 42 0 0 0
13 20 0 0 0
14 48 0 0 0
15
16 2. COMPRESSION with direction d:
17
18 0 0 0 0
19 0 0 0 0
20 0 0 0 0
21 124 0 0 0
22
23
```

```
PS C:\Users\Pau Tong Lin Xu\Desktop\CPPProject
s\projects\project8> cd "c:\Users\Pau Tong Lin
Xu\Desktop\CPPProjects\projects\project8\" ; if
($?) { g++ project8.cpp -o project8 } ; if
($?) { .\project8 }
SIZE=4, RAND=100
INITIAL MATRIX:
2 4 4 4
8 16 2 16
8 2 2 8
16 8 16 8
1. COMPRESSION with direction 1:
14 0 0 0
42 0 0 0
20 0 0 0
48 0 0 0
2. COMPRESSION with direction d:
0 0 0 0
0 0 0 0
0 0 0 0
124 0 0 0
PS C:\Users\Pau Tong Lin Xu\Desktop\CPPProject
s\projects\project8>
```

如上图所示，当 'config.txt' 里面有 'LOG = FILENAME' 但 'BOX' 不等于 'TRUE'，则程序把结果输出到terminal和名为 FILENAME 的文件里（在此测试里文件叫 'log.txt'）。该输出不用边框围绕数字。

为了得到上面的结果，我先初始化了一个叫做 isLog 的 boolean 全局变量。接下来，我创建了 output() 函数。如果 isLog 为 true，则每次使用 output() 它都会分别用 'cout' 和 'fout' 把参数 'text' 输出到terminal 和 'log.txt' 里。

```
void output(string text, bool endl, int width){ // outputs in terminal (and log file)
// the function will output in log file if global 'isLog' variable is true
// for formatting purposes, the function also takes in the width and line skips as parameters

if(width != 0){
cout << setw(width);
fout << setw(width);
}
if (isLog) {
cout << text;
fout << text;
if (endl){
cout << endl;
fout << endl;
}
}
else{
cout << text;
if (endl){
cout << endl;
}
}
}
```

后面每次输出字符串都用 output()。

```

void normalOutput(){ // outputs the matrix in non-box format
    for (int i = 0; i<SIZE ; i++){
        output("----", false, 0);
    }
    output("", true, 0);
    for (int r =0 ; r<SIZE ; r++){
        for (int c=0; c<SIZE ; c++){
            if (matrix[r][c] == -1){
                output(" ", false, 3);
            }
            else{
                output(to_string(matrix[r][c]), false, 3);
            }
            output(" ", false, 0);
        }
        output("", true, 0);
    }
    for (int i = 0; i<SIZE ; i++){
        output("----", false, 0);
    }
    output("", true, 0);
}

```

## 测试 3 - 无LOG, 有BOX

The screenshot shows a C++ IDE with three panels: a configuration file, the main source code, and a terminal window.

**config.txt:**

```

1 SIZE = 4
2 RAND = 100
3 BOX = TRUE
4

```

**project8.cpp:**

```

21 // so I use a 2d pointer to allocate dynamic memory to my matrix
22 int SIZE;
23 int** matrix;
24
25 string LOG;
26 ofstream fout;
27
28 bool isLog = false;
29 bool BOX = false;
30
31 int getRandom(int max){ // returns an integer between 1 and
32 // The generation seed is created in the main() function
33     return (rand()*max/RAND_MAX)+1;
34 }
35
36 void output(string text, bool endl, int width){ // output
37 // the function will output in log file if global 'isLog'
38 // for formatting purposes, the function also takes in t
39
40 if(width != 0){
41     cout << setw(width);
42     fout << setw(width);
43 }
44 if (isLog) {
45     cout << text;
46     fout << text;
47     if (endl){
48         cout << endl;
49         fout << endl;
50     }
51 }
52 else{
53     cout << text;
54     if (endl){
55         cout << endl;
56     }
57 }
58 }
59
60 void normalOutput(){ // outputs the matrix in non-box format
61     for (int i = 0; i<SIZE ; i++){
62         output("----", false, 0);
63     }
64     output("", true, 0);
65     for (int r =0 ; r<SIZE ; r++){
66         for (int c=0; c<SIZE ; c++){
67             if (matrix[r][c] == -1){
68                 output(" ", false, 3);
69             }
70             else{
71                 output(to_string(matrix[r][c]), false, 3);
72             }
73             output(" ", false, 0);
74         }
75         output("", true, 0);
76     }
77     for (int i = 0; i<SIZE ; i++){
78         output("----", false, 0);
79     }
80     output("", true, 0);
81 }

```

**Terminal Output:**

```

PS C:\Users\Pau Tong Lin Xu\Desktop\CPPProject
s\projects\project8> cd "c:\Users\Pau Tong Lin
Xu\Desktop\CPPProject\projects\project8\" ;
if ($?) { g++ project8.cpp -o project8 } ; if
($?) { .\project8 }
SIZE=4, RAND=100
INITIAL MATRIX:
+-----+
| 16| 2| 8| 8|
+-----+
| 4| 16| 4| 4|
+-----+
| 8| 8| 16| 2|
+-----+
| 8| 16| 4| 2|
+-----+
1. COMPRESSION with direction 1:
+-----+
| 34| 0| 0| 0|
+-----+
| 28| 0| 0| 0|
+-----+
| 34| 0| 0| 0|
+-----+
| 30| 0| 0| 0|
+-----+
2. COMPRESSION with direction d:
+-----+
| 0| 0| 0| 0|
+-----+
| 0| 0| 0| 0|
+-----+
| 0| 0| 0| 0|
+-----+
| 126| 0| 0| 0|
+-----+
PS C:\Users\Pau Tong Lin Xu\Desktop\CPPProject
s\projects\project8>

```

如上图所示，当 'config.txt' 里面没有 'LOG = FILENAME' 但 'BOX' 等于 'TRUE'，则程序只把结果输出到terminal和里。该输出使用边框围绕数字。可以看到输出第三个矩阵是，最大的数字变成了三位数，则矩阵里所有单元的大小从2个字符变成了3个字符。

为了输出用边框围绕数字的矩阵，我写了下面的 boxedOutput() 函数。

```

void boxedOutput(){ // outputs the matrix in box format
    int width = getMax();

    for (int r = 0 ; r<SIZE ; r++){
        for (int c=0; c<SIZE ; c++){
            output("+", false, 0);
            for (int i = 0 ; i<width ; i++){
                output("-", false, 0);
            }
            output("+", true, 0);

            for (int c=0; c<SIZE ; c++){
                output("|", false, 0);
                if (matrix[r][c] == -1){
                    output("*", false, width);
                }
                else{
                    output(to_string(matrix[r][c]), false, width);
                }
            }
            output("|", true, 0);
        }

        for (int c=0; c<SIZE ; c++){
            output("+", false, 0);
            for (int i = 0 ; i<width ; i++){
                output("-", false, 0);
            }
        }
        output("+", true, 0);
    }
}

```

每次调用 boxedOutput(), 它会再调用下面的 getMax() 函数来找矩阵里字符最多的单元。而输出的矩阵里每个单元的大小是 getMax() 返回的数值。

```

int getMax(){ // gets the maximum length of a values in the matrix
    int max = 0;
    for (int i = 0 ; i<SIZE ; i++){
        for (int j = 0 ; j<SIZE ; j++){
            if (to_string(matrix[i][j]).length() > max){
                max = to_string(matrix[i][j]).length();
            }
        }
    }
    return max;
}

```

## 测试 4 - 有LOG, 有BOX

The screenshot shows a C++ IDE with three main panels. The left panel displays 'config.txt' with the following content:

```
1 SIZE = 4
2 RAND = 100
3 BOX = TRUE
4 LOG = log.txt
```

The middle panel shows 'project8.cpp' with the following code:

```
21 // so I use a 2d pointer to allocate dynamic memory to my matrix
22 int SIZE;
23 int** matrix;
24
25 string LOG;
26 ofstream fout;
27
28 bool islog = false;
29 bool BOX = false;
30
31 int getRandom(int max){ // returns an integer between 1 and
32 // The generation seed is created in the main() function
33 return (rand()*max/RAND_MAX)+1;
34 }
35
36 void output(string text, bool endl, int width){ // output
37 // the function will output in log file if global 'islog'
38 // for formatting purposes, the function also takes in t
39
40 if(width != 0){
41 cout << setw(width);
42 fout << setw(width);
43 }
44 if (islog) {
45 cout << text;
46 fout << text;
47 if (endl){
48 cout << endl;
49 fout << endl;
50 }
51 }
52 else{
53 cout << text;
54 if (endl){
55 cout << endl;
56 }
57 }
58 }
59
60 void normalOutput(){ // outputs the matrix in non-box format
61 for (int i = 0; i<SIZE; i++){
62 output("----", false, 0);
```

The right panel shows the output of the program, which includes the initial matrix and the results of two compression steps:

```
1 SIZE=4, RAND=100
2 INITIAL MATRIX:
3 -----
4 [16] 2[16] 4]
5 -----
6 [ 4] 2[ 2] 4]
7 -----
8 [16] 0[16] 0]
9 -----
10 [ 4] 4[ 2] 8]
11 -----
12 1. COMPRESSION with direction 1:
13 -----
14 [38] 0[ 0] 0]
15 -----
16 [12] 0[ 0] 0]
17 -----
18 [48] 0[ 0] 0]
19 -----
20 [18] 0[ 0] 0]
21 -----
22 2. COMPRESSION with direction d:
23 -----
24 [ 0] 0[ 0] 0]
25 -----
26 [ 0] 0[ 0] 0]
27 -----
28 [ 0] 0[ 0] 0]
29 -----
30 [116] 0[ 0] 0]
31 -----
32
```

The terminal panel shows the command prompt output, which matches the output in the right panel.

如上图所示，当 'config.txt' 里面有 'LOG = FILENAME' 且 'BOX' 等于 'TRUE'，则程序把结果输出到terminal和名为 FILENAME 的文件里(在此测试里文件叫 'log.txt')。该输出使用边框围绕数字。

程序会调用 outputstep() 来输出每个步骤进行后留下的新的矩阵。若boolean全局变量 BOX 为真，则用边框输出矩阵，否则就用 normalOutput() 输出。

```
void outputstep(){ // function chooses whether to use box format depending on global 'BOX' variable
    if(BOX){
        boxedOutput();
    }
    else{
        normalOutput();
    }
}
```