CS205 C/C++ Programming - Lab ASSIGNMENT 3

Name : 杨睦圳（Yang Muzhen）

SID ： 11813010

Part 1 - Analysis

Question 1

In this question, we need to find out all the places which are not dangerous.

Use a two-dimensional array which stores the boolean value to show the map, and the point with "true" value means dangerous. We can use the following method to check if a place is dangerous:

(x_p,y_p) : the point's place, (x_b, y_b) : the bullet's initial place, (dx,dy) : the direction of bullet.

$a = x_p - x_b$, $b = y_p - y_b$.

If dx = 0, then the point is dangerous when a = 0 and b / dy ≥ 0

If dy = 0, then the point is dangerous when b = 0 and a / dx ≥ 0

Else, i.e. dx ≠ 0 and dy ≠ 0, the point is dangerous when a / dx = b / dy ≥ 0

Question 2

In this question, we need to get an m * n matrix with elements from 1 to m * n in anticlockwise spiral order, starting from the top right corner.

Consider a two-dimensional array as the matrix. Initialize it with all the elements 0, which means all the points are accessible. Use a big loop in which there are four other loops to generate the matrix. Use two variable `int cnt = 0` and `bool flag = true` . The four loops are move left, down, right and up in order. For each loop, while the next point is accessible, set flag to "true", move into the next point, set the value of it to `cnt` and add `cnt` 1. Everytime we start a new iteration of the outer loop, set the flag to "false". The outer loop is finished when at the end of an iteration the value of `flag` is still false, which means there is no any point accessible now, i.e. we have generated the matrix.

Question 3

The structure we define to store the blocks' information is

```
struct block{
    string hex_start;
    string hex_end;
    unsigned int start;
    unsigned int end;
    string name;
};
```

From the "Blocks.txt", we can get hex_start, hex_end and name of the block. Two functions `toNum()` and `toDecimal()` are used to get the integer value of start and end of a block according to hex_start, hex_end.

Define the function `search()` which takes a character as a parameter and return the index of its block in the array of blocks.

Count the number of occurance of the character in different blocks from the input file and then print out the blocks which most characters belong.

Part 2 - Code

The program is worked on cygwin

Question 1

```
#include <iostream>
using namespace std;

void check(bool *map[], int *bullet[], int n, int m, int k){
    int i,j,d;
    for(i = 0; i < n; i++){
        for(j = 0; j < m; j++){
            for(d = 0; d < k; d++){
                int x = bullet[d][0];
                int y = bullet[d][1];
                int dx = bullet[d][2];
                int dy = bullet[d][3];

                if(dx != 0 && dy != 0){
                    int a = (i - x) / dx;
                    int b = (j - y) / dy;
                    if(a == b && a >= 0){
                        map[i][j] = true;
                        break;
                    }
                }else if(dx != 0){
                    int a = (i - x) / dx;
                    int b = j - y;
                    if(a >= 0 && b == 0){
                        map[i][j] = true;
                        break;
                    }
                }else if(dy != 0 ){
                    int a = i - x;
                    int b = (j - y) / dy;
                    if(b >= 0 && a == 0){
                        map[i][j] = true;
                        break;
                    }
                }
            }
        }
    }
}
```

```
bool inputCorrect(int a, int b){
    if(a < -1 || a > 1 || b < -1 || b > 1)
        return false;

    if(a == 0 && b == 0)
        return false;

    return true;
}

int main(){
    int n,m,k,i,j;
    cin >> n;
    cin >> m;
    cin >> k;

    if(n <= 0 || m <= 0){
        cout << "Please input the correct data!" << endl;
        return 0;
    }

    bool **map = new bool*[n];
    for(i = 0; i < n; i++){
        map[i] = new bool [m];
        for(j = 0; j < m; j++)
            map[i][j] = false;
    }

    int **bullet = new int*[k];
    for(i = 0; i < k; i++){
        bullet[i] = new int[4];
        cin >> bullet[i][0];
        cin >> bullet[i][1];
        cin >> bullet[i][2];
        cin >> bullet[i][3];
        if(!inputCorrect(bullet[i][2],bullet[i][3])){
            cout << "Please input the correct direction!" << endl;
            return 0;
        }
    }

    check(map, bullet, n, m, k);

    int cnt = 0;
    for(i = 0; i < n; i++){
        for(j = 0; j < m; j++){
            if(!map[i][j])
                cnt++;
        }
    }
    cout << cnt << endl;

    for(i = 0; i < n; i++)
        delete[] map[i];
    delete[] map;

    for(i = 0; i < m; i++)
        delete[] bullet[i];
    delete[] bullet;

    return 0;
}
```

Question 2

```
#include <iostream>
#define inMap(i,j,n,m) 0 <= i && i < n && 0 <= j && j < m
using namespace std;

void print_map(int *map[], int n, int m){
    int i,j;
    for(i = 0; i < n; i++){
        for(j = 0; j < m; j++)
            cout << map[i][j] << "\t";
        cout << endl;
    }
    cout << endl;
}

void fill_map(int *map[], int n, int m){
    int i = 0, j = m - 1, cnt = 1;
    map[i][j] = 1;
    bool flag = true;
    while(flag){
        flag = false;
        while(inMap(i,j - 1,n,m) && map[i][j - 1] == -1){
            j = j - 1;
            cnt++;
            flag = true;
            map[i][j] = cnt;
        }
        while(inMap(i + 1,j,n,m) && map[i + 1][j] == -1){
            i = i + 1;
            cnt++;
            flag = true;
            map[i][j] = cnt;
        }
        while(inMap(i,j + 1,n,m) && map[i][j + 1] == -1){
            j = j + 1;
            cnt++;
            flag = true;
            map[i][j] = cnt;
        }
        while(inMap(i - 1,j,n,m) && map[i - 1][j] == -1){
            i = i - 1;
            cnt++;
            flag = true;
            map[i][j] = cnt;
        }
    }
}

int main(){
    int n, m;
    cin >> n;
    cin >> m;

    if(n <= 0 || m <= 0){
        cout << "Please input the correct size!" << endl;
        return 0;
    }

    int i,j;
    int **map = new int*[n];
    for(i = 0; i < n; i++){
        map[i] = new int[m];
        for(j = 0; j < m; j++)
            map[i][j] = -1;
    }

    fill_map(map, n, m);

    print_map(map, n, m);

    for(i = 0; i < n; i++)
        delete[] map[i];
    delete[] map;

    return 0;
}
```

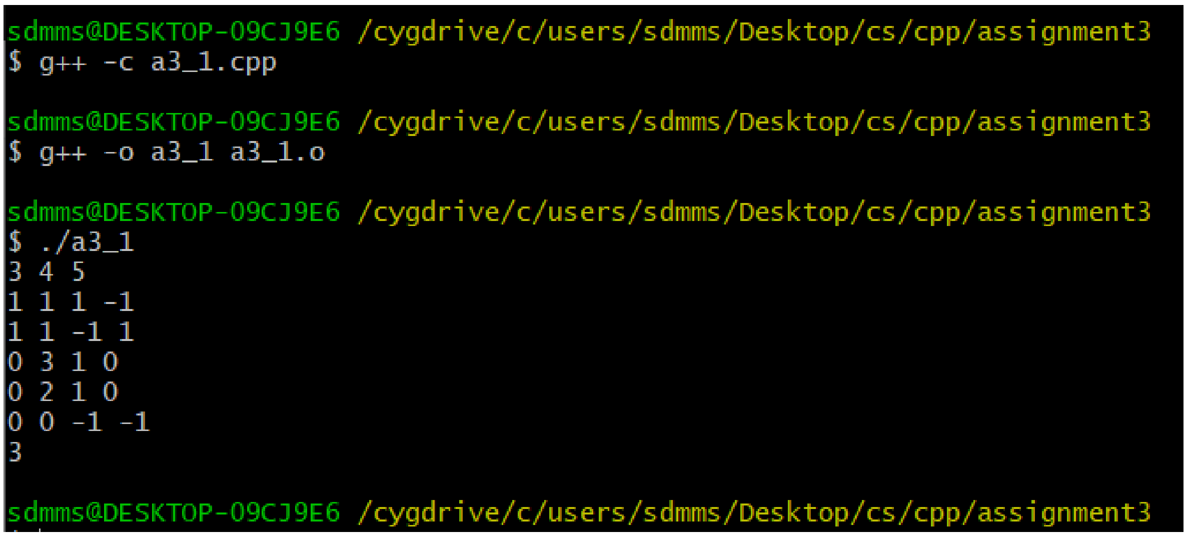
Question 3

```
#include <iostream>
#include <fstream>
#include <vector>
#include "utf8.c"
using namespace std;

struct block{
    string hex_start;
    string hex_end;
    unsigned int start;
    unsigned int end;
    string name;
};

unsigned int toDecimal(char a){
    a = toupper(a);
    unsigned int index = (unsigned int) a;
    if(index >= 48 && index <= 57)
        return index - 48;
    else
        return index - 55;
}

unsigned int toNum(string str){
    unsigned int i, result = 0;
    for(i = 0; i < str.size(); i++)
        result = result * 16 + toDecimal(str[i]);
    return result;
}

bool read_block_info(block arr[], int *n, string fileName){
    ifstream fin(fileName);
    if(!fin.good()){
        cout << "File missing! Program exit!"<< endl;
        return false;
    }
    string in;
    while(getline(fin,in)){
        if(in[0] == '#' || in.size() <= 1)
            continue;

        int space = in.find_first_of(' ');
        arr[*n].name = in.substr(space + 1);

        int dot = in.find_first_of('.');
        arr[*n].hex_start = in.substr(0,dot);
        arr[*n].start = toNum(arr[*n].hex_start);
        arr[*n].hex_end = in.substr(dot + 2, dot);
        arr[*n].end = toNum(arr[*n].hex_end);
        (*n)++;
    }
    return true;
}

int search(block arr[], unsigned char* p, int *len, int n){
    unsigned int cp = utf8_to_codepoint(p, len);
    if (cp) {
        int i;
        for(i = 0; i < n; i++){
            if(cp < arr[i].start)
                return i - 1;
        }
        return n - 1;
    } else {
        return -1;
    }
}

void print_max_block(const int cnt[], block arr[], int n, int no_block){

    int i, max = 0;
    for(i = 0; i < n; i++){
        if(cnt[i] > max)
            max = cnt[i];
    }

    if(max <= 0)
        cout << "Please give the valid data!" << endl;
    else{
        vector<int> temp;
        for(i = 0; i < n; i++){
            if(cnt[i] == max){
                temp.push_back(i);
            }
        }
        for(i = 0; i < temp.size(); i++)
            cout << arr[temp[i]].name << endl;
    }
}

int main(int argc, char** fileName){

    block arr[300];
    int n = 0;
    if(!read_block_info(arr, &n, "Blocks.txt"))
        return 0;

    int no_block = 0;
    int cnt[300];

    char a[1000000];
    while (cin.getline(a,1000000)){
        auto* p = (unsigned char *)a;
        int len = 0;
        while (*p) {
            int index = search(arr,p, &len, n);
            if (index != -1) {
                cnt[index]++;
                p += len;
            } else {
                no_block++;
                p++;
            }
        }
    }

    print_max_block(cnt, arr, n, no_block);

    return EXIT_SUCCESS;
}
```

Part 3 - Result & Verification

Question 1

Compilation:



```
sdmms@DESKTOP-09CJ9E6 /cygdrive/c/users/sdmms/Desktop/cs/cpp/assignment3
$ g++ -c a3_1.cpp

sdmms@DESKTOP-09CJ9E6 /cygdrive/c/users/sdmms/Desktop/cs/cpp/assignment3
$ g++ -o a3_1 a3_1.o

sdmms@DESKTOP-09CJ9E6 /cygdrive/c/users/sdmms/Desktop/cs/cpp/assignment3
$ ./a3_1
3 4 5
1 1 1 -1
1 1 -1 1
0 3 1 0
0 2 1 0
0 0 -1 -1
3

sdmms@DESKTOP-09CJ9E6 /cygdrive/c/users/sdmms/Desktop/cs/cpp/assignment3
```

Test Case #1:

```
Input:
5 5 8
2 0 1 -1
3 3 -1 0
0 1 -1 -1
2 3 -1 0
2 0 -1 0
4 0 -1 0
3 1 -1 1
4 1 1 1
Output:
11
```

Test Case #2:

```
Input :
1 3 1
0 0 0 1
Output:
0
```

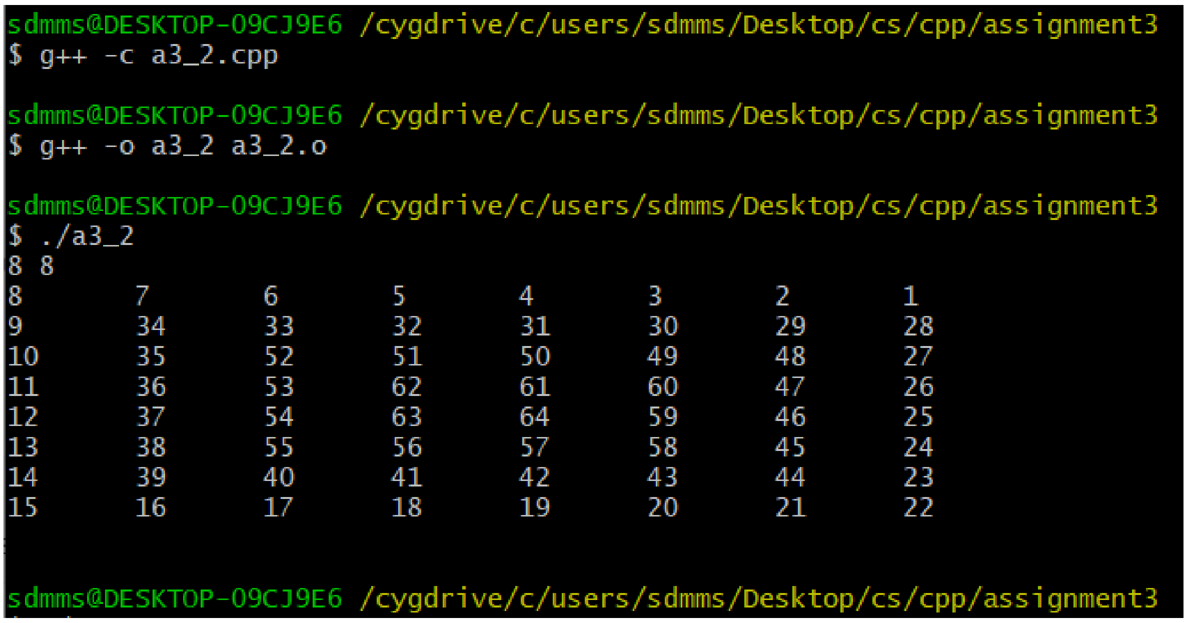
Test Case #3:

```
Input:
10 10 20
4 9 -1 -1
0 5 1 1
4 9 1 0
5 1 1 0
8 6 1 -1
8 4 1 1
4 8 1 1
3 9 0 1
5 4 -1 1
7 9 -1 1
```

```
2 3 1 1
9 3 1 1
9 9 -1 1
5 1 -1 1
8 3 -1 1
0 3 -1 0
6 1 1 1
1 8 -1 -1
1 0 -1 0
2 3 1 -1
Output:
50
```

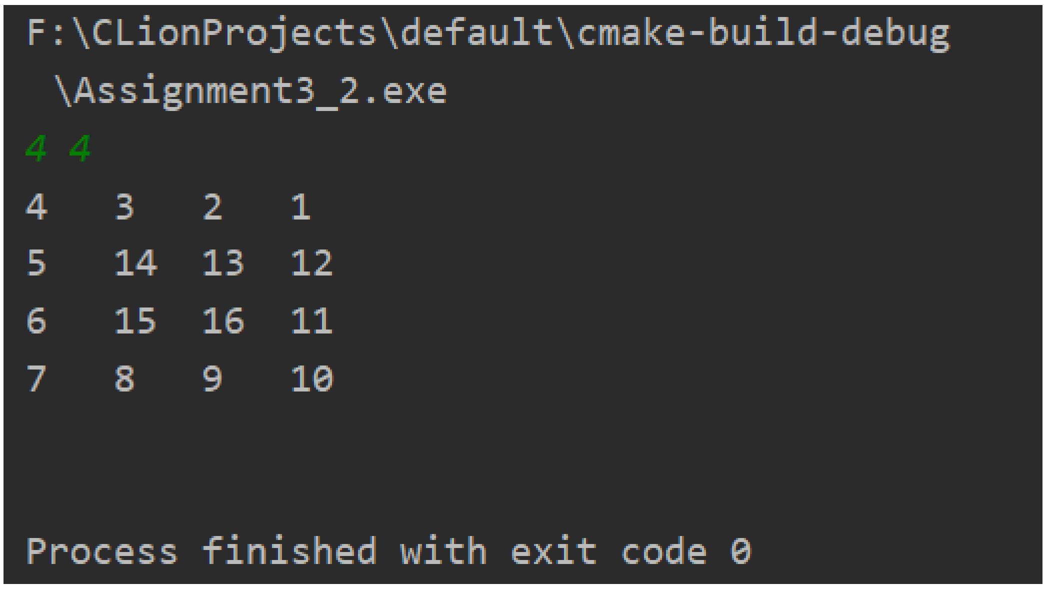
Question 2

Compilation:



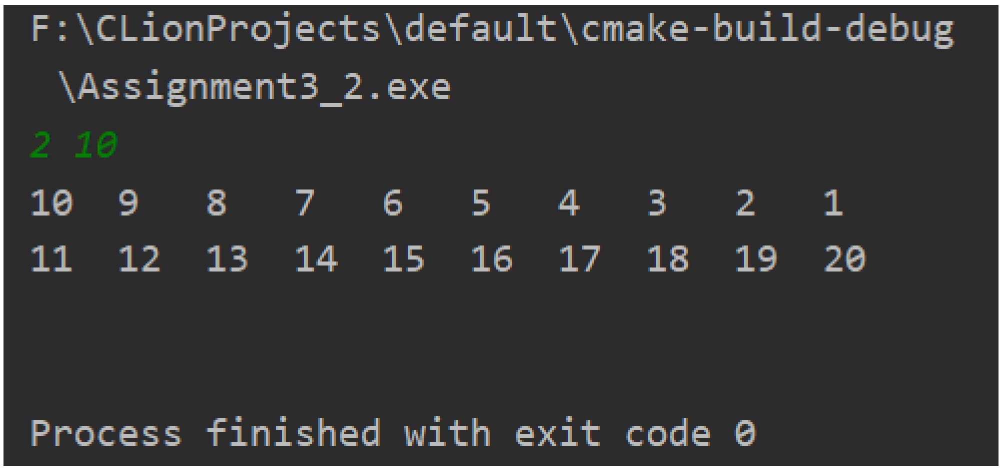
```
sdmms@DESKTOP-09CJ9E6 /cygdrive/c/users/sdmms/Desktop/cs/cpp/assignment3
$ g++ -c a3_2.cpp

sdmms@DESKTOP-09CJ9E6 /cygdrive/c/users/sdmms/Desktop/cs/cpp/assignment3
$ g++ -o a3_2 a3_2.o

sdmms@DESKTOP-09CJ9E6 /cygdrive/c/users/sdmms/Desktop/cs/cpp/assignment3
$ ./a3_2
8 8
8       7       6       5       4       3       2       1
9       34      33      32      31      30      29      28
10      35      52      51      50      49      48      27
11      36      53      62      61      60      47      26
12      37      54      63      64      59      46      25
13      38      55      56      57      58      45      24
14      39      40      41      42      43      44      23
15      16      17      18      19      20      21      22


sdmms@DESKTOP-09CJ9E6 /cygdrive/c/users/sdmms/Desktop/cs/cpp/assignment3
```

Test Case #1:



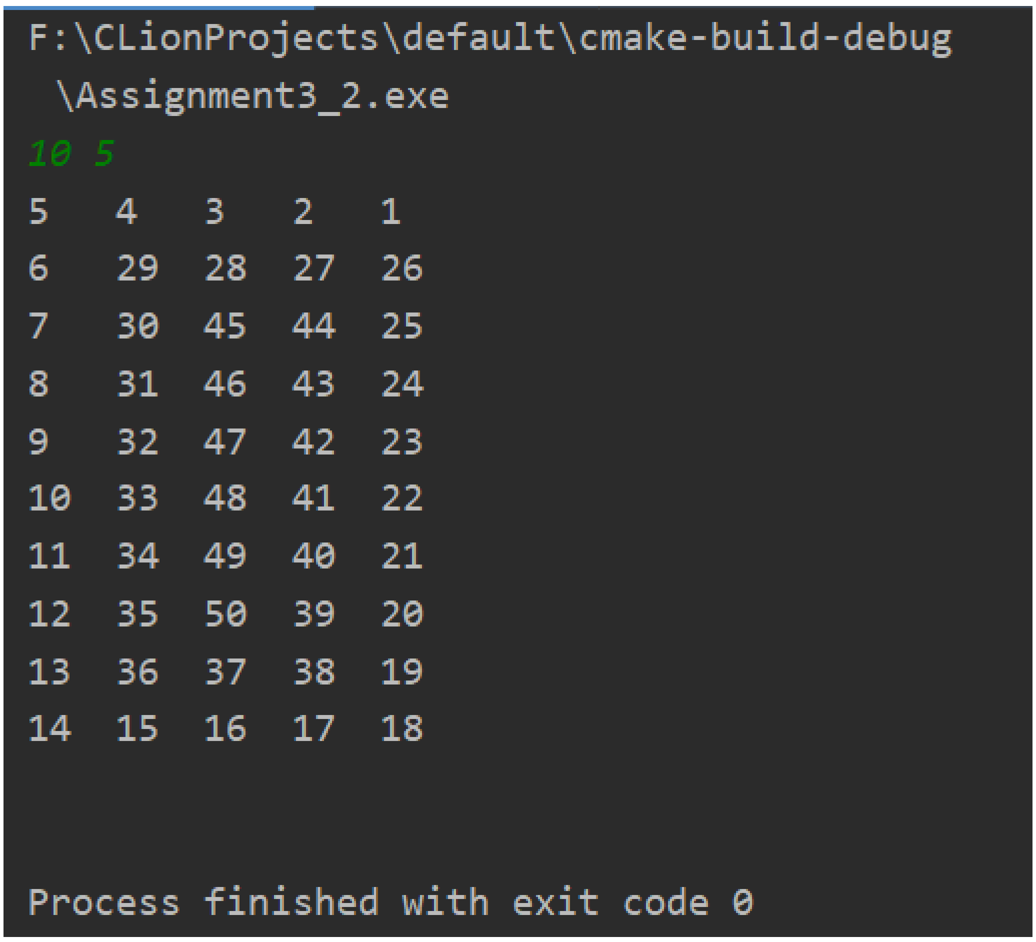
```
F:\CLionProjects\default\cmake-build-debug
 \Assignment3_2.exe
4 4
4   3   2   1
5   14  13  12
6   15  16  11
7   8   9   10


Process finished with exit code 0
```

Test Case #2:



```
F:\CLionProjects\default\cmake-build-debug
 \Assignment3_2.exe
2 10
10  9   8   7   6   5   4   3   2   1
11  12  13  14  15  16  17  18  19  20


Process finished with exit code 0
```

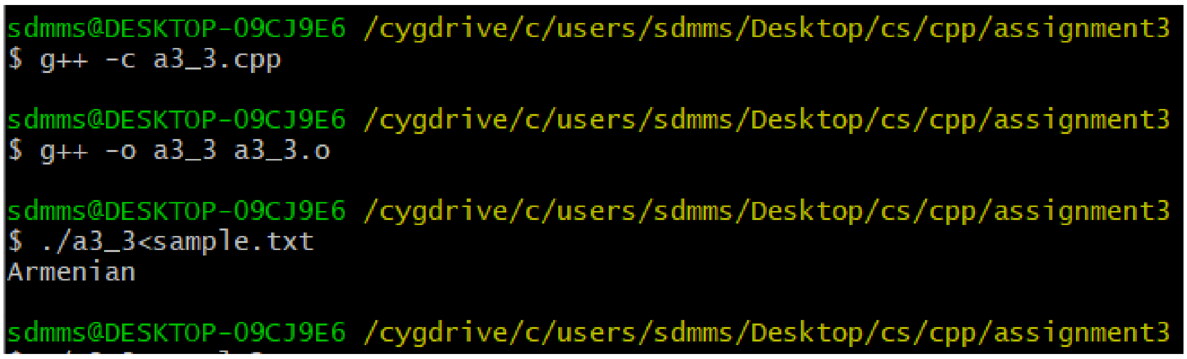
Test Case #3:



```
F:\CLionProjects\default\cmake-build-debug
 \Assignment3_2.exe
10 5
5   4   3   2   1
6   29  28  27  26
7   30  45  44  25
8   31  46  43  24
9   32  47  42  23
10  33  48  41  22
11  34  49  40  21
12  35  50  39  20
13  36  37  38  19
14  15  16  17  18


Process finished with exit code 0
```

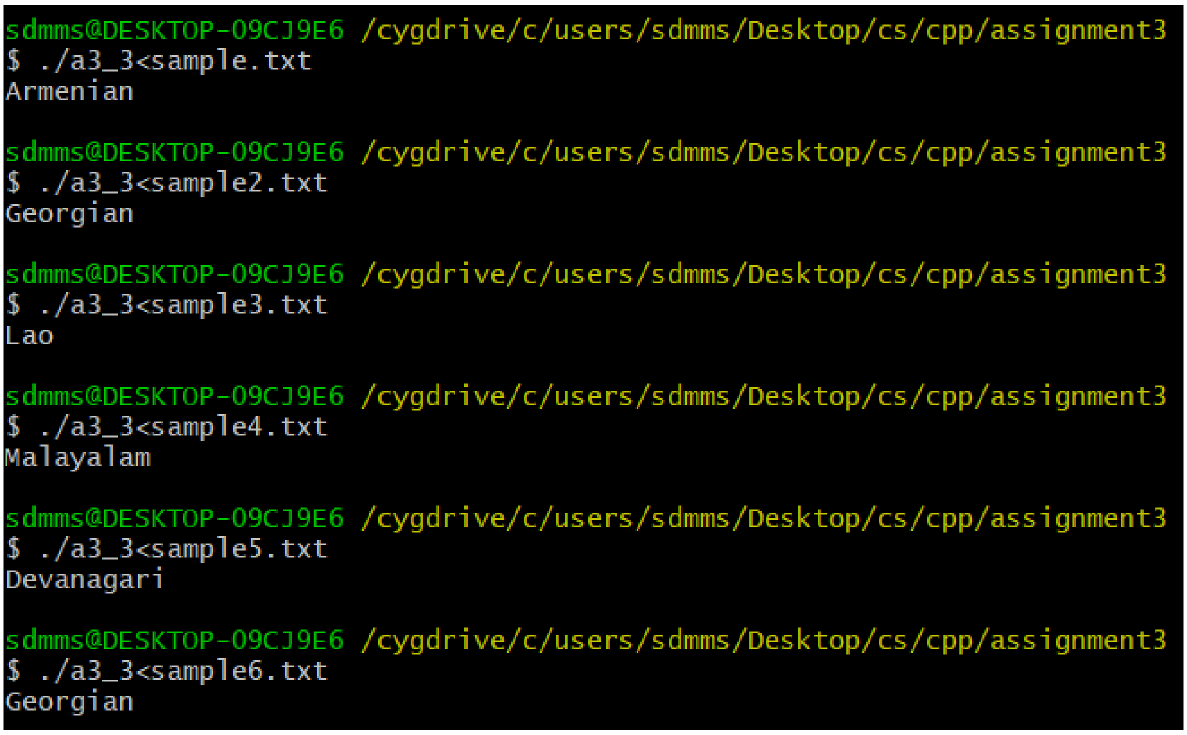
Question 3

Compilation:



```
sdmms@DESKTOP-09CJ9E6 /cygdrive/c/users/sdmms/Desktop/cs/cpp/assignment3
$ g++ -c a3_3.cpp

sdmms@DESKTOP-09CJ9E6 /cygdrive/c/users/sdmms/Desktop/cs/cpp/assignment3
$ g++ -o a3_3 a3_3.o

sdmms@DESKTOP-09CJ9E6 /cygdrive/c/users/sdmms/Desktop/cs/cpp/assignment3
$ ./a3_3<sample.txt
Armenian

sdmms@DESKTOP-09CJ9E6 /cygdrive/c/users/sdmms/Desktop/cs/cpp/assignment3
```

Test Case:



```
sdmms@DESKTOP-09CJ9E6 /cygdrive/c/users/sdmms/Desktop/cs/cpp/assignment3
$ ./a3_3<sample.txt
Armenian

sdmms@DESKTOP-09CJ9E6 /cygdrive/c/users/sdmms/Desktop/cs/cpp/assignment3
$ ./a3_3<sample2.txt
Georgian

sdmms@DESKTOP-09CJ9E6 /cygdrive/c/users/sdmms/Desktop/cs/cpp/assignment3
$ ./a3_3<sample3.txt
Lao

sdmms@DESKTOP-09CJ9E6 /cygdrive/c/users/sdmms/Desktop/cs/cpp/assignment3
$ ./a3_3<sample4.txt
Malayalam

sdmms@DESKTOP-09CJ9E6 /cygdrive/c/users/sdmms/Desktop/cs/cpp/assignment3
$ ./a3_3<sample5.txt
Devanagari

sdmms@DESKTOP-09CJ9E6 /cygdrive/c/users/sdmms/Desktop/cs/cpp/assignment3
$ ./a3_3<sample6.txt
Georgian
```

Part 4 - Difficulties & Solutions

In question 3, when I use `p++;` to move to the next character, I am unable to get the expected result in most of times. After leaning about the utf8 code and understanding the template code, I know that the length of the character is different. The solution is adding the corresponding length of a character to move to the next character.