

BÁO CÁO BUỔI THỰC HÀNH SỐ 3

Bộ Môn Kỹ Thuật Lập Trình



Sinh viên: Trịnh Thăng Việt Anh – 20210067

Giáo viên hướng dẫn: ThS. Lê Thị Hoa

Bài thực hành số 3 – Tuần 13

Mục Lục

Bài thực hành số 3 – Tuần 13	2
Bài 3.1. Dãy Lucas được định nghĩa bởi $L_n = L_{n-1} + L_{n-2}$ với $L_0 = 2$, $L_1 = 1$. Hãy viết hàm tính số Lucas thứ n.....	6
Hình 3.1.1 Code bài 1.....	7
Hình 3.1.1 Test case bài 1	7
Bài 3.2. Trên bàn cờ vua kích thước $n*n$ có một quân mã đang ở ô $(1, 1)$. Hãy đưa ra một dãy các di chuyển của mã sao cho mỗi ô trên bàn cờ đều được đi qua đúng 1 lần ($(1, 1)$ được xem là đã đi qua).....	7
Hình 3.2.1 Code bài 2.....	7
Hình 3.2.2 Test case bài 2	9
Bài 3.3. Một người xuất phát tại thành phố 1, muốn đi thăm tất cả các thành phố khác, mỗi thành phố đúng 1 lần và quay về 1. Chi phí để đi từ thành phố i sang thành phố j là c_{ij} . Hãy tìm tổng chi phí nhỏ nhất có thể.....	9
Hình 3.3.1 Code bài 3.....	9
Hình 3.3.2 Test case bài 3	10
Bài 3.4. Cho dãy a có n phần tử. Một dãy con của a là dãy thu được bằng cách xóa đi một số phần tử của a và giữ nguyên thứ tự các phần tử còn lại (có thể không xóa phần tử nào). Hãy tìm dãy con tăng dài nhất của a. Dữ liệu vào:	10
Hình 3.4.1 Code bài 4.....	11
Hình 3.4.2 Test case bài 4	11
Bài 3.5. Tính hệ số tổ hợp $C(n, k)$	12
Hình 3.5.1 Code bài 5.....	12
Hình 3.5.2 Test case bài 5	14
Bài 3.6. Tìm ước chung lớn nhất của hai số nguyên a, b cho trước.	14
Hình 3.6.1 Code bài 6.....	14
Hình 3.6.1 Test case bài 6	14
Bài 3.7. Sử dụng phương pháp khử đệ quy bằng stack, hãy liệt kê các xâu nhị phân độ dài n không có k bit 1 nào liên tiếp.....	14
Hình 3.7.1 Code bài 7.....	15
Hình 3.7.2 Test case bài 7	17
Bài 3.8 Cân đĩa Bạn đang muốn kiểm tra xem một vật có trọng lượng MM như người ta nói hay không. Có một cái cân bằng và n cân. Quá trình thử nghiệm nặng nề. Hãy chỉ ra một cách cân thỏa mãn. Quy cách in ra đã được tích hợp trong mã nguồn dưới đây.....	17

Hình 3.8.1 Code bài 8.....	18
Hình 3.8.2 Code bài 8 (Tiếp)	18
Hình 3.8.3 Test mẫu bài 8	19
Bài 3.9. Một y tá cần lập lịch làm việc trong Ngày, mỗi ngày chỉ có thể là làm việc hay nghỉ ngơi. Một lịch làm việc là tốt nếu không có hai ngày nghỉ nào liên tiếp và mọi chuỗi ngày tối đa làm việc liên tiếp đều có số ngày thuộc đoạn $[K1, K2]$. Hãy liệt kê tất cả các cách lập lịch tốt, với mỗi lịch in ra trên một dòng một xâu nhị phân độ dài n với bit 0/1 tương ứng là nghỉ/làm việc. Các xâu phải được in ra theo thứ tự từ điển.....	19
Hình 3.9.1 Code bài 9.....	20
Hình 3.9.2 Code bài 9(Tiếp).	21
Hình 3.9.3 Code bài 9(Tiếp)	21
Hình 3.9.4 Testcase 1 bài 9	21
Hình 3.9.5 Testcase 2 bài 9	22
Hình 3.9.6 Testcase 3 bài 9	22
Bài 3.10 Khoảng cách Hamming giữa hai xâu cùng độ dài là số vị trí mà ký tự tại vị trí đó là khác nhau trên hai xâu. Cho S là xâu gồm n ký tự 0. Hãy liệt kê tất cả các xâu nhị phân độ dài n , có khoảng cách Hamming với S bằng H . Các xâu phải được liệt kê theo thứ tự từ điển	23
Hình 3.10.1 Code bài 10.....	24
Hình 3.10.2 Code bài 10(Tiếp)	24
Hình 3.10.3 Testcase 1 bài 10	25
Hình 3.10.4 Testcase ví dụ bài 10	25
Bài 3.11 Superior là một hòn đảo tuyệt đẹp với n địa điểm chụp ảnh và các đường một chiều nối các điểm chụp ảnh với nhau. Đoàn khách tham quan có r người với sở thích chụp ảnh khác nhau. Theo đó, mỗi người sẽ đưa ra danh sách các địa điểm mà họ muốn chụp. Bạn cần giúp mỗi người trong đoàn lập lịch di chuyển sao cho đi qua các điểm họ yêu cầu đúng một lần, không đi qua điểm nào khác, bắt đầu tại điểm đầu tiên và kết thúc tại điểm cuối cùng trong danh sách mà họ đưa ra, và có tổng khoảng cách đi lại là nhỏ nhất.....	26
Hình 3.11.1 Code bài 11.....	27
Hình 3.11.2 Code bài 11 (Tiếp)	27
Hình 3.11.3 Code bài 11 (Tiếp)	28
Hình 3.11.4 Test case 1 bài 11	28
Hình 3.11.5 Test case 2 bài 11	29
Bài 3.12 Cho đồ thị vô hướng G , hãy đếm số đường đi đi qua k cạnh và không đi qua đỉnh nào quá một lần.....	29
Hình 3.12.1 Code bài 12.....	30
.....	31
Hình 3.12.2 Code bài 12 (Tiếp)	31
Hình 3.12.3 Testcase 1 bài 12	32

Hình 3.12.4 Testcase 2 bài 12	32
Hình 3.12.5 Testcase 3 bài 12	33
Hình 3.12.6 Testcase 4 bài 12	33
Hình 3.12.7 Testcase 5 bài 12	34
Hình 3.12.8 Testcase 6 bài 12	34
Hình 3.12.9 Testcase 7 bài 12	34
Hình 3.12.10 Testcase 8 bài 12	35
Hình 3.12.11 Testcase 9 bài 12	35

Bài 3.1. Dãy Lucas được định nghĩa bởi $L_n = L_{n-1} + L_{n-2}$ với $L_0 = 2$, $L_1 = 1$. Hãy viết hàm tính số Lucas thứ n..

Answer: (penalty regime: 10, 20, ... %)

```
1 /* Bài 3.1 - Tuần 13
2 Võ Hung Đức - 4836 -744469 -20241 */
3 int lucas(int n_viduc) {
4     if (n_viduc == 0) return 2; // L0 = 2
5     if (n_viduc == 1) return 1; // L1 = 1
6     return lucas(n_viduc - 1) + lucas(n_viduc - 2); // Ln = Ln-1 + Ln-2
7 }
```

```
/* Bài 3.1 - Tuần 13
Võ Hung Đức - 4836 -744469 -20241 */

#include<bits/stdc++.h> // Thư viện chuẩn C++ bao gồm các thư viện cần thiết
using namespace std; // Sử dụng không gian tên std để tránh khái niệm std:: trước các hàm

// Hàm tính số Lucas tại vị trí n_viduc
int lucas(int n_viduc) {
    if (n_viduc == 0) return 2; // Nếu n_viduc là 0, trả về 2 (số Lucas đầu tiên)
    if (n_viduc == 1) return 1; // Nếu n_viduc là 1, trả về 1 (số Lucas thứ hai)
    // Để tính số Lucas tại vị trí n_viduc bằng tổng của hai số Lucas trước đó
    return lucas(n_viduc - 1) + lucas(n_viduc - 2);
}

int main() {
    int n_viduc; // Khai báo biến n_viduc để lưu vị trí số Lucas cần tính
    std::cin >> n_viduc; // Nhận giá trị cho n_viduc từ bàn phím
    std::cout << lucas(n_viduc); // In ra số Lucas tại vị trí n_viduc
    return 0; // Kết thúc chương trình
}
```

Code đã comment

	Test	Expected	Got	
✓	cout << lucas(5);	11	11	✓
✓	cout << lucas(10);	123	123	✓
✓	cout << lucas(30);	1860498	1860498	✓

Passed all tests! ✓

Correct

Marks for this submission: 10.00/10.00.

Hình 3.1.1 Test case bài 1

Bài 3.2. Trên bàn cờ vua kích thước $n \times n$ có một quân mã đang ở ô $(1, 1)$. Hãy đưa ra một dãy các di chuyển của mã sao cho mỗi ô trên bàn cờ đều được đi qua đúng 1 lần ($\text{ô } (1, 1)$ được xem là đã đi qua).

Answer: (penalty regime: 10, 20, ... %)

```

1 /* Bài 3.2 - Tuần 13
2 Vi Hung Duc - 4836 - 744469 -20241 */
3 #include <iostream>
4 using namespace std;
5
6 int n_viduc; // Kích thước bàn cờ (n x n)
7 int X_viduc[100], Y_viduc[100]; // Lưu tọa độ các bước di chuyển của quân mã
8 int mark_viduc[100][100]; // Đánh dấu vị trí các ô mà quân mã đã di chuyển qua
9
10 // Mảng hx_viduc, hy_viduc mô tả 8 vị trí quân mã có thể di chuyển kể từ vị trí hiện tại
11 const int hx_viduc[] = {1, 1, 2, 2, -1, -1, -2, -2}; // Các thay đổi về tọa độ x
12 const int hy_viduc[] = {2, -2, 1, -1, 2, -2, 1, -1}; // Các thay đổi về tọa độ y
13
14 // In ra dãy các di chuyển tìm được
15 void print_sol(){
16     for (int j = 1; j <= n_viduc * n_viduc; ++j)
17         printf("%d %d\n", X_viduc[j], Y_viduc[j]); // In tọa độ từng bước di chuyển
18     exit(0); // Kết thúc chương trình khi đã tìm được một giải pháp
19 }
20
21 // Thuật toán quay lui
22 void TRY(int k){
23     for(int i = 0; i < 8; i++){ // Duyệt qua tất cả 8 hướng di chuyển của quân mã
24         int xx_viduc = X_viduc[k-1] + hx_viduc[i]; // Tính toán tọa độ mới x
25         int yy_viduc = Y_viduc[k-1] + hy_viduc[i]; // Tính toán tọa độ mới y
26         // Kiểm tra xem tọa độ mới có hợp lệ không
27         if(!mark_viduc[xx_viduc][yy_viduc] && (1 <= xx_viduc) && (xx_viduc <= n_viduc) && (1 <= yy_viduc) && (yy_viduc <= n_viduc)){
28             mark_viduc[xx_viduc][yy_viduc] = 1; // Đánh dấu ô đã được quân mã di chuyển qua
29             xx_viduc = yy_viduc; // Cập nhật tọa độ mới vào mảng
30             yy_viduc = xx_viduc; // Cập nhật tọa độ mới vào mảng
31             TRY(k+1); // Tiếp tục di chuyển đến bước tiếp theo
32         } else TRY(k+1); // Tiếp tục di chuyển đến bước tiếp theo
33         mark_viduc[xx_viduc][yy_viduc] = 0; // Quay lại trạng thái trước đó (backtrack)
34     }
35 }
36
37
38 int main(){
39     cin >> n_viduc; // Nhập kích thước bàn cờ
40     mark_viduc[1][1] = 1; // Đánh dấu ô  $(1,1)$  đã được di chuyển qua
41     X_viduc[1] = Y_viduc[1] = 1; // Lưu tọa độ bắt đầu
42     TRY(2); // Bắt đầu thuật toán quay lui từ bước thứ 2
43     return 0; // Kết thúc chương trình
44 }
```

Hình 3.2.1 Code bài 2

	Input	Expected	Got	
✓	5	(1 1) (2 3) (3 5) (5 4) (4 2) (2 1) (3 3) (1 4) (2 2) (4 1) (5 3) (4 5) (2 4) (1 2) (3 1) (5 2) (4 4) (2 5) (1 3) (3 2) (5 1) (4 3) (5 5) (3 4) (1 5)	(1 1) (2 3) (3 5) (5 4) (4 2) (2 1) (3 3) (1 4) (2 2) (4 1) (5 3) (4 5) (2 4) (1 2) (3 1) (5 2) (4 4) (2 5) (1 3) (3 2) (5 1) (4 3) (5 5) (3 4) (1 5)	✓
✓	6	(1 1) (2 3) (3 5) (4 3) (5 5) (6 3) (5 1) (3 2) (5 3) (6 1) (4 2) (2 1) (1 3) (2 5) (4 6) (6 5) (4 4) (5 6) (6 4) (5 2) (3 1) (1 2) (3 3) (4 5) (6 6) (5 4) (6 2) (4 1) (2 2) (1 4) (2 6) (3 4) (1 5) (3 6) (2 4) (1 6)	(1 1) (2 3) (3 5) (4 3) (5 5) (6 3) (5 1) (3 2) (5 3) (6 1) (4 2) (2 1) (1 3) (2 5) (4 6) (6 5) (4 4) (5 6) (6 4) (5 2) (3 1) (1 2) (3 3) (4 5) (6 6) (5 4) (6 2) (4 1) (2 2) (1 4) (2 6) (3 4) (1 5) (3 6) (2 4) (1 6)	✓
✓	7	(1 1) (2 3) (3 5) (4 7) (5 5) (6 7) (7 5) (6 3) (7 1) (5 2) (6 4) (7 6) (5 7) (6 5) (7 7) (5 6) (4 4) (3 6) (2 4) (1 6) (3 7) (4 5) (5 3) (7 2) (5 1) (3 2) (1 3) (2 1) (4 2) (6 1) (7 3) (5 4) (6 6) (7 4) (6 2) (4 3) (3 1) (1 2) (3 3) (4 1) (2 2) (1 4) (2 6) (3 4) (1 5) (2 7) (4 6) (2 5) (1 7)	(1 1) (2 3) (3 5) (4 7) (5 5) (6 7) (7 5) (6 3) (7 1) (5 2) (6 4) (7 6) (5 7) (6 5) (7 7) (5 6) (4 4) (3 6) (2 4) (1 6) (3 7) (4 5) (5 3) (7 2) (5 1) (3 2) (1 3) (2 1) (4 2) (6 1) (7 3) (5 4) (6 6) (7 4) (6 2) (4 3) (3 1) (1 2) (3 3) (4 1) (2 2) (1 4) (2 6) (3 4) (1 5) (2 7) (4 6) (2 5) (1 7)	✓

Passed all tests! ✓

Hình 3.2.2 Test case bài 2

Bài 3.3. Một người xuất phát tại thành phố 1, muốn đi thăm tất cả các thành phố khác, mỗi thành phố đúng 1 lần và quay về 1. Chi phí để đi từ thành phố i sang thành phố j là c_{ij} . Hãy tìm tổng chi phí nhỏ nhất có thể

Answer: (penalty regime: 10, 20, ... %)

```

1 /* Bài 3.3 - Tuần 13
2 Vi Hung Duc - 4836 -744469 -20241 */
3 #include <iostream>
4 #include<limits.h>
5 using namespace std;
6 #define MAX 100
7
8 int n_viduc, c[MAX][MAX]; // số thành phố và ma trận chi phí
9 int cmin_viduc = INT_MAX; // chi phí đi lại nhỏ nhất giữa hai thành phố khác nhau
10 int best_viduc = INT_MAX; // tổng chi phí nhỏ nhất cần tìm, ban đầu đặt bằng giá trị vô cùng lớn INT_MAX = 2^31-1
11 int curr_viduc; // tổng chi phí tối thời điểm hiện tại
12 int mark_viduc[MAX]; // đánh dấu những thành phố đã đi
13 int x_viduc[MAX]; // lưu giữ các thành phố đã đi
14
15 // Đọc dữ liệu vào
16 void input(){
17     cin >> n_viduc;
18     for (int i = 1; i <= n_viduc; ++i)
19         for (int j = 1; j <= n_viduc; ++j){
20             cin >> c[i][j]; // nhập dữ liệu
21             if (c[i][j] > 0) cmin_viduc = min(cmin_viduc, c[i][j]); // tìm chi phí nhỏ nhất giữa 2 thành phố trong quá trình nhập
22         }
23 }
24
25 // Thuật toán quay lui
26 void TRY(int k){
27     for(int i = 2; i <= n_viduc; i++){
28         if(!mark_viduc[i]){
29             x_viduc[k] = i;
30             mark_viduc[i] = 1; // đánh dấu trạng thái hiện tại đã được thăm
31             curr_viduc += c[x_viduc[k-1]][i]; // cập nhật tổng chi phí
32             if(k == n_viduc){ // nếu duyệt hết rồi
33                 best_viduc = min(best_viduc, curr_viduc + c[i][1]); // lấy chi phí nhỏ nhất (chi phí + quay trở lại về 1)
34             }else if(curr_viduc + cmin_viduc * (n_viduc-k+1) < best_viduc){ // nếu tổng chi phí còn lại (tham lam, nhân cmin_viduc với cả
35                 TRY(k+1); // thì đệ quy tăng k lên 1
36             }
37             mark_viduc[i] = 0; // nếu đệ quy xong mà vẫn chưa tìm được lời giải thì quay lui
38             curr_viduc -= c[x_viduc[k-1]][i]; // cập nhật lại tổng chi phí
39         }
40     }
41 }
42
43 int main() {
44     input();
45     x_viduc[1] = 1;
46     TRY(2);
47     cout << best_viduc;
48     return 0;
49 }
50

```

Hình 3.3.1 Code bài 3

	Input	Expected	Got	
✓	4 0 2 1 3 4 0 1 2 2 1 0 3 3 4 2 0	7	7	✓
✓	6 0 2 1 3 7 3 4 0 1 2 8 5 2 1 0 3 6 9 3 4 2 0 2 3 1 7 3 9 0 4 2 1 4 5 6 0	11	11	✓
✓	8 0 2 1 3 7 3 2 7 4 0 9 2 8 5 13 2 2 4 0 3 6 9 1 5 3 3 2 0 2 3 7 3 2 7 3 9 0 4 1 9 2 1 4 5 6 0 5 4 1 6 7 2 4 5 0 10 12 2 3 1 4 5 6 0	14	14	✓

Passed all tests! ✓

Hình 3.3.2 Test case bài 3

Bài 3.4. Cho dãy a có n phần tử. Một dãy con của a là dãy thu được bằng cách xóa đi một số phần tử của a và giữ nguyên thứ tự các phần tử còn lại (có thể không xóa phần tử nào). Hãy tìm dãy con tăng dài nhất của a.

Dữ liệu vào:

Dòng 1: Chứa số nguyên n ($1 \leq n \leq 1000$)

Dòng 2: Chứa n số nguyên a_1, a_2, \dots, a_n ($|a_i| \leq 10^9$)

Kết quả:

Dòng đầu tiên chứa độ dài dãy con tăng dài nhất

Dòng thứ 2 chứa chỉ số các phần tử được chọn vào dãy con đó

Nếu có nhiều dãy con tăng dài nhất, in ra dãy bất kỳ trong số đó

Code đã comment

```

<global> lis(int i) : int
Start here X Bai3.12.cpp X Bai3.8.cpp X Bai3.4.cpp X Bai3.5.cpp X Bai3.7.cpp X Bai3.9.cpp X Bai3.10.cpp X Bai3.11.cpp X
1 // C++ 3.4 - Tuan 13
2 // Vi Hung Duc - 744469 - 20241
3 #include <iostream> // Đầu file cần các đầu vào đầu ra cần có để
4 #include <cstring> // Đầu file cần các đầu vào đầu ra cần có để
5
6 using namespace std; // Để dùng không gian tên std để truy cập biến và hàm std::
7
8 int a_viduc[1000], n_viduc; // Đầu file cần khai báo biến a_viduc có kích thước 1000 và biến n_viduc là số lượng
9 int mem_viduc[1000]; // Số lượng biến cần khai báo trước khi sử dụng con trỏ
10
11 // Đầu file cần khai báo biến a_viduc với giá trị -1
12 void init()
13 {
14     memset(mem_viduc, -1, sizeof(mem_viduc)); // Gán tất cả các biến trong mem_viduc thành -1
15 }
16
17 // Quá trình thuật toán
18 // Duyệt qua từng phần tử của mảng a_viduc và tìm phần tử có giá trị là -1
19 int lis(int i)
20 {
21     if(mem_viduc[i] != -1) // Nếu giá trị của mem_viduc[i] không bằng -1
22         return mem_viduc[i]; // Trả về giá trị đó và thoát
23
24     int result = 1; // Đầu tiên giá trị con số lớn nhất là 1 (tính từ phần tử a_viduc[1])
25     for(int j = 0; j < i; j++) // Duyệt qua các phần tử sau phần tử a_viduc[1]
26         if(a_viduc[j] < a_viduc[i]) // Nếu phần tử sau nhỏ hơn phần tử hiện tại
27             result = max(result, 1 + lis(j)); // Gán giá trị của phần tử con số lớn nhất
28
29     mem_viduc[i] = result; // Gán giá trị của phần tử con số lớn nhất
30     return result; // Trả về giá trị này
31 }
32
33 // Quá trình lối gốc
34 void trace(int i)
35 {
36     for(int j = 0; j < i; j++) // Duyệt qua các phần tử sau phần tử a_viduc[1]
37         if (a_viduc[j] < a_viduc[i] && mem_viduc[i] == 1 + mem_viduc[j]) // Nếu phần tử sau bằng phần tử hiện tại
38             trace(j); // Duyệt qua phần tử sau và tiếp tục
39         break; // Khi tìm thấy phần tử sau bằng phần tử hiện tại
40 }
41
42 cout << a_viduc[i] << " "; // In ra giá trị của a_viduc[i]
43
44 int main()
45 {
46     init(); // Đầu file cần khai báo biến a_viduc
47     cin >> n_viduc; // Đầu file cần khai báo biến n_viduc
48     for(int i = 0; i < n_viduc; i++) cin >> a_viduc[i]; // Đầu file cần khai báo biến a_viduc
49     int res_viduc = 1, pos_viduc = 0; // Đầu file cần khai báo biến res_viduc và pos_viduc
50     for(int i = 1; i < n_viduc; i++)
51         if (res_viduc < lis(i)) // Nếu giá trị của phần tử con số lớn nhất hiện tại
52             res_viduc = lis(i); // Gán giá trị của phần tử con số lớn nhất
53             pos_viduc = i; // Gán giá trị của phần tử con số lớn nhất
54
55     cout << res_viduc << endl; // In ra giá trị của res_viduc
56     trace(pos_viduc); // Đầu file cần khai báo biến pos_viduc
57     return 0; // Đầu file cần khai báo biến
58 }

```

Logs & others

Code::Blocks X	Search results X	Cccc X	Build log X	Build messages X	CppCheck/Vera++ X	CppCheck/Vera++ messages X	Cscope X	Debugger X
File	Line	Message	==== Build file: "no target" in "no project" (compiler: unknown) ===					

```

1 /* Bài 3.4 – Tuần 13
2 Vi Hung Duc - 4826 - 744469 - 20241 */
3 #include <iostream>
4 #include <cstring>
5
6 using namespace std;
7 int a_viduc[1000], n_viduc;
8 int mem_viduc[1000]; // mảng ghi nhớ lời giải các bài toán con đã được giải
9
10 void init(){
11     memset(mem_viduc, -1, sizeof(mem_viduc));
12 }
13
14 // Quy hoạch động,
15 // Hàm lis(i) trả về độ dài dãy con tăng dài nhất kết thúc bởi a_viduc[i]
16 int lis(int i) {
17     if(mem_viduc[i] == -1){
18         return mem_viduc[i];
19     }
20     int result = 1;
21     for(int j = 0; j < i; j++){
22         if(a_viduc[j] < a_viduc[i]){
23             result = max(result, 1+lis(j));
24         }
25     }
26     mem_viduc[i] = result;
27     return result;
28 }
29
30
31 // Truy vết lời giải
32 void trace(int i){
33     for(int j = 0; j < i; j++){
34         if (a_viduc[j] < a_viduc[i] && mem_viduc[i] == 1 + mem_viduc[j]){
35             trace(j);
36             break;
37         }
38     }
39     cout << a_viduc[i] << " ";
40 }
41
42 int main(){
43     init();
44     cin >> n_viduc;
45     for(int i = 0; i < n_viduc; i++) cin >> a_viduc[i];
46     int res_viduc = 1, pos_viduc = 0;
47     for(int i = 1; i < n_viduc; i++){
48         if (res_viduc < lis(i)){
49             res_viduc = lis(i);
50             pos_viduc = i;
51         }
52     }
53     cout << res_viduc << endl;
54     trace(pos_viduc);
55     return 0;
56 }
57

```

Hình 3.4.1 Code bài 4

Input	Expected	Got	
✓ 6 2 1 5 4 3 6	3 2 5 6	3 2 5 6	✓
✓ 10 2 1 5 4 3 6 3 -6 9 10	5 2 5 6 9 10	5 2 5 6 9 10	✓
✓ 28 2 -10 -8 5 4 3 6 3 -6 9 10 12 2 5 25 9 15 30 -100 45	10 -10 -8 5 6 9 10 12 25 30 45	10 -10 -8 5 6 9 10 12 25 30 45	✓

Passed all tests! ✓

Hình 3.4.2 Test case bài 4

Bài 3.5. Tính hệ số tổ hợp C(n, k)

Answer: (penalty regime: 10, 20, ... %)

```

1 /* Bài 3.5 - Tuần 13
2 Vi Hung Duc - 4836 - 744469 -20241 */
3 #include <iostream> // Thư viện cho nhập xuất cơ bản
4
5 using namespace std;
6
7 // Hàm tính hệ số nhị thức C(n, k) bằng đệ quy
8 int binom(int n_viduc, int k_viduc) {
9     if (k_viduc > n_viduc) return 0; // Nếu k lớn hơn n, trả về 0
10    if (k_viduc == 0) return 1; // C(n, 0) luôn bằng 1
11    // Đệ quy tính C(n, k) = C(n-1, k) + C(n-1, k-1)
12    return binom(n_viduc-1, k_viduc) + binom(n_viduc-1, k_viduc-1);
13 }
14
15 int c_viduc[1000][1000]; // Mảng 2 chiều để lưu trữ các giá trị C(n, k)
16 int binom2(int n_viduc, int k_viduc){
17     // Khởi tạo giá trị cho C(n, 0) = 1
18     // Khởi tạo giá trị cho C(n, 0) = 1
19     for(int i = 0; i <= n_viduc; i++){
20         c_viduc[i][0] = 1; // C(n, 0) = 1 cho mọi n
21     }
22     for(int j = 1; j <= i; j++){ // Tính C(n, k) dựa trên các giá trị đã tính trước đó
23         c_viduc[i][j] = c_viduc[i-1][j] + c_viduc[i-1][j-1]; // C(n, k) = C(n-1, k) + C(n-1, k-1)
24     }
25 }
26
27 return c_viduc[n_viduc][k_viduc]; // Trả về giá trị C(n, k) đã tính
28 }
29
30 int main(){
31     int n_viduc; // Biến để lưu số lượng n
32     cin >> n_viduc; // Nhập vào giá trị n_viduc từ người dùng
33     // Vòng lặp để in ra các hệ số nhị thức bằng phương pháp đệ quy
34     for (int n_viduc = 1; n_viduc <= m_viduc; ++n_viduc){
35         for (int k_viduc = 0; k_viduc <= n_viduc; ++k_viduc)
36             printf("%d ", binom(n_viduc, k_viduc)); // In ra C(n, k) bằng đệ quy
37             printf("\n"); // Xuống dòng sau mỗi hàng
38     }
39     // Vòng lặp để in ra các hệ số nhị thức bằng phương pháp khử đệ quy
40     for (int n_viduc = 1; n_viduc <= m_viduc; ++n_viduc){
41         for (int k_viduc = 0; k_viduc <= n_viduc; ++k_viduc)
42             printf("%d ", binom2(n_viduc, k_viduc)); // In ra C(n, k) bằng khử đệ quy
43             printf("\n"); // Xuống dòng sau mỗi hàng
44     }
45 }

```

Precheck

Check

Hình 3.5.1 Code bài 5

	Input	Expected	Got
✓	4	1 1 1 2 1 1 3 3 1 1 4 6 4 1 1 1 1 2 1 1 3 3 1 1 4 6 4 1	1 1 1 2 1 1 3 3 1 1 4 6 4 1 1 1 1 2 1 1 3 3 1 1 4 6 4 1
✓	10	1 1 1 2 1 1 3 3 1 1 4 6 4 1 1 5 10 10 5 1 1 6 15 20 15 6 1 1 7 21 35 35 21 7 1 1 8 28 56 70 56 28 8 1 1 9 36 84 126 126 84 36 9 1 1 10 45 120 210 252 210 120 45 10 1 1 1	1 1 1 2 1 1 3 3 1 1 4 6 4 1 1 5 10 10 5 1 1 6 15 20 15 6 1 1 7 21 35 35 21 7 1 1 8 28 56 70 56 28 8 1 1 9 36 84 126 126 84 36 9 1 1 10 45 120 210 252 210 120 45 10 1 1 1
My Courses ▾ English (en) ▾			
Time left 22:54:00			

Passed all tests! ✓

*Hình 3.5.2 Test case bài 5***Bài 3.6.** Tìm ước chung lớn nhất của hai số nguyên a, b cho trước.

Answer: (penalty regime: 10, 20, ... %)

```

1 */* Bài 3.6 - Tuần 13
2 Võ Hùng Đức - 4836 -744469 -20241 */
3 #include <iostream> // Thư viện cho nhập xuất dữ liệu
4
5 using namespace std; // Sử dụng không gian tên std để tránh phải viết std:: trước các hàm
6
7 // Hàm tính ước số chung lớn nhất (GCD) bằng đệ quy
8 int gcd(int a_viduc, int b_viduc){
9     if (b_viduc == 0) return a_viduc; // Nếu b_viduc bằng 0, trả về a_viduc
10    return gcd(b_viduc, a_viduc % b_viduc); // Gọi đệ quy với b_viduc và a_viduc chia cho b_viduc
11 }
12
13 // Hàm tính GCD bằng phương pháp lặp
14 int gcd2(int a_viduc, int b_viduc){
15     while(b_viduc){ // Khi b_viduc khác 0
16         a_viduc %= b_viduc; // Cập nhật a_viduc là a_viduc chia cho b_viduc
17         swap(a_viduc, b_viduc); // Hoán đổi giá trị của a_viduc và b_viduc
18     }
19     return a_viduc; // Trả về a_viduc là GCD
20 }
21
22 int main() {
23     int a_viduc, b_viduc; // Khai báo hai biến nguyên a_viduc và b_viduc
24     cin >> a_viduc >> b_viduc; // Nhập giá trị cho a_viduc và b_viduc từ bàn phím
25     cout << gcd(a_viduc, b_viduc) << endl << gcd2(a_viduc, b_viduc); // In ra GCD bằng cả hai phương pháp
26     return 0; // Kết thúc chương trình
27 }
```

Precheck

Check

Hình 3.6.1 Code bài 6

	Input	Expected	Got	
✓	50 35	5 5	5 5	✓
✓	217 413	7 7	7 7	✓

Passed all tests! ✓

*Hình 3.6.1 Test case bài 6***Bài 3.7.** Sử dụng phương pháp khử đệ quy bằng stack, hãy liệt kê các xâu nhị phân độ dài n không có k bit 1 nào liên tiếp**Dữ liệu vào:**Một dòng duy nhất chứa hai số nguyên n, k ($1 \leq k \leq n \leq 20$)**Kết quả:**

Với mỗi xâu tìm được, in ra n ký tự trên một dòng, các ký tự cách nhau bởi dấu cách. Các xâu cần được liệt kê theo thứ tự từ điển.

Answer: (penalty regime: 10, 20, ... %)

Time left 22:40:30

```

1 /* Bài 3.7 - Tuần 13
2 Vi Hung Duc - 4836 -744469 -20241 */
3 #include <bits/stdc++.h>
4
5 using namespace std;
6
7 struct state{
8     int i, j, old_L;
9     //# constructor
10    state(int _i = 0, int _j = 0, int _L = 0):
11        i(_i), j(_j), old_L(_L){}
12 };
13
14 int main() {
15     int n_viduc, k_viduc;
16     cin >> n_viduc >> k_viduc;
17     int x_viduc[n_viduc+1];
18     stack<state> s_viduc;
19     //# number of consecutive suffix 1
20     int L_viduc = 0;
21     s_viduc.push(state(1, 0));
22     while (!s_viduc.empty()){
23         state &top = s_viduc.top(); //lấy ra phần tử đầu tiên của stack
24         //# if a new binary sequence is found
25         if (top.i > n_viduc){
26             // Nếu top.i vượt quá chiều dài của xâu nhị phân (top.i > n_viduc), in ra xâu nhị phân đã tạo và pop trạng thái đó khỏi stack
27             for (int i = 1; i <= n_viduc; ++i)
28                 cout << x_viduc[i] << " ";
29             s_viduc.pop();
30             continue;
31         }
32
33         //# Khiết đê quy
34         if(top.j > 0)
35             L_viduc = top.old_L;
36         if(top.j >1){
37             s_viduc.pop();
38             continue;
39         }
40         if(L_viduc + 1 < k_viduc || top.j == 0)
41         {
42             x_viduc[top.i] = top.j;
43             top.old_L = L_viduc;
44             L_viduc = top.j ? L_viduc + 1 : 0;
45             s_viduc.push(state(top.i + 1, 0));
46         }
47         ++top.j;
48     }
49     return 0;
50 }
51
52 }
53
54

```

Hình 3.7.1 Code bài 7

Hình 3.7.2 Test case bài 7

Bài 3.8 Cân đĩa Bạn đang muốn kiểm tra xem một vật có trọng lượng MM như người ta nói hay không. Có một cái cân bằng và n cân. Quá trình thử nghiệm nặng nề. Hãy chỉ ra một cách cân thỏa mãn. Quy cách in ra đã được tích hợp trong mã nguồn dưới đây.

Dữ liệu mẫu:

6 10

7 1 2 3 4 5

Kết quả mẫu:

$$-1 + 2 + 3 + 4 + 5 = 10$$

The screenshot shows the Code::Blocks IDE interface. The top window displays the source code for `Bai3.8.cpp`. The code implements a solution for the `n-pan balance` problem using a stack-based approach. It reads input values from `n_video` and prints them to `cout`. The bottom window shows the build log, which indicates that no target was found in the project.

```

1 // Bai 3.8 - Tuan 13
2 // Vi Hung Duc - 4836 - 744469 - 20241 *
3 #include <iostream>
4 using namespace std;
5
6 // Kien thuc cua thanh de lam tanh bao nhieu (1, 5)
7 struct state {
8     int l, r; // l: canh oai vao, r: canh khac cua vao
9     state(int _l = 0, int _r = 0) { l = _l, r = _r } // Ham khieu lac
10 };
11
12 int main() {
13     int n_video, M_video; // so luong canh, so luong luong canh
14     cin >> n_video >> M_video; // Nhieu oai Luong canh va so luong luong canh
15     int m_video[n_video+1]; // Mang luu truong luong canh oai vao
16     for (int i = 0; i <= n_video; ++i) cin >> m_video[i]; // Nhieu luu truong luong canh vao
17     int x[n_video+1]; // Mang luu truong thanh chon vao (1: chon, -1: khong chon)
18     state s; // Kien thuc cua thanh de lam tanh bao nhieu
19
20     // sum = 0 // Banh luu oai luong luong de chon
21     s.push(state(1, -1)); // Banh trang thai dau tien vao ngan han
22     while (1) {
23         if (s.size() == 1) { // Banh trang thai cuoi cung
24             state top = s.top(); // Banh trang thai cuoi cung
25             if (top.l > n_video) // Nhu cau hien tra loi cua canh oai
26                 if (sum == M_video) // Nhu oai luong luong bieu M_video
27                     for (int i = 0; i <= n_video; ++i) // Tu lai oai oai
28                         if (x[i] == 1) cout << m_video[i]; // Yet khieu chon
29                         if (x[i] == -1) cout << ' ' << m_video[i]; // Yet khieu chon
30                 cout << endl << M_video; // In ra so luong luong
31             exit(0); // Bat kinh chuong trinh
32         }
33         s.pop(); // Xoa luong luong tron chon
34         continue; // Tiep tuc luon lai
35     }
36     if (top.l > -1) // nhu cau hien trang thai de chon
37         sum = m_video[top.l] - x[top.l]; // oai khac oai luong luong
38     if (top.l > 1) // nhu cau hien cau trang thai cho vat bien tai
39         s.pop(); // Xoa luong luong tron chon
40         continue; // Tiep tuc luon lai
41     }
}

```

Logs & others

File Line Message

*** Build file: "no targets" in "no project" (compiler: unknown) ***

Hình 3.8.1 Code bài 8

```

Bai8.cpp - CodeBlocks 20.03
File Edit View Search Project Build Debug Fortran wxSmith Tools Tools+ Plugins Doxygen Settings Help
<global> main() int
Start here X Bai8.cpp X Bai8.cpp X
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
    int main() {
        int n_videu, M_videu; // số lượng và số lượng trọng lượng cần tìm
        cin >> n_videu >> M_videu; // Nhập số lượng và số lượng trọng lượng
        int m_videu[n_videu]; // Mảng lưu trữ trọng lượng của các vật
        for (int i = 0; i < n_videu; i++) cin >> m_videu[i]; // Nhập trọng lượng cần tìm
        int x0, y0; // Tọa độ ban đầu của vật cần tìm (x0, y0) -1, [nhập x0, y0]
        stack<state> s; // Sử dụng stack để lưu trạng thái
        // sum of selected weights
        int sum = 0; // Biên lưu tổng trọng lượng đã chọn
        s.push(state(-1, -1)); // Đầu tiên là tọa độ ban đầu của vật cần tìm
        while (!s.empty()) { // Lặp truy cập tất cả các trạng thái
            state top = s.top(); // Lấy trạng thái trên cùng của ngăn xếp
            if (top.x == n_videu) // Nếu đã tìm ra tọa độ của vật cần tìm
                if (top.y == M_videu) // Nếu số lượng trọng lượng bằng M_videu
                    cout << "Tìm thấy cách đi đúng" << endl;
                else // Nếu số lượng trọng lượng không đúng
                    cout << "Không tìm thấy cách đi" << endl;
            else { // Nếu chưa tìm thấy
                if (top.y > -1) // Nếu đã chọn trạng thái trước đó
                    sum += m_videu[top.y]; // Tính tổng trọng lượng
                if (top.y > 0) { // Nếu đã chọn vật có cách đi xa hơn
                    s.pop(); // Xóa trạng thái trên cùng
                    continue; // Tiếp tục tìm kiếm
                }
                if (top.y > 1) { // Nếu đã chọn vật có cách đi xa hơn
                    s.pop(); // Xóa trạng thái trên cùng
                    continue; // Tiếp tục tìm kiếm
                }
                if (top.y == 0) { // Nếu đã chọn vật có cách đi xa nhất
                    s.pop(); // Xóa trạng thái trên cùng
                    cout << "Không tìm thấy cách đi" << endl;
                }
                else { // Nếu chưa tìm thấy
                    if (top.y < M_videu) { // Nếu chưa tìm thấy
                        s.push(state(top.x, top.y + 1)); // Thêm vào ngăn xếp
                        ++top.y; // Tăng chỉ số trạng thái
                    }
                }
            }
            cout << "->" << endl; // Nhập khoảng cách cách nhau mỗi lần in -1
        }
        return 0; // Kết thúc chương trình
    }

```

Logs & others

CodeBlocks X Search results X Ccc X Build log X CppCheck/Vera++ X CppCheck/Vera++ messages X Cscope X Debugger X Doxygen X Fortran info X Closed files list X Thread search X

File Line Message

== Build file: "no targets" in "no project" (compiler: unknown) ==

C:\Users\DELL\Desktop\KTLBuoi3\Bai8.cpp

21C C#

Windows (CR+LF) UTF-8 Line 17, Col 69, Pos 809 Insert Read/Write default ENG US SOS PM 12/6/2024

Hình 3.8.2 Code bài 8 (Tiếp)

```

Bai3.8.cpp - CodeBlocks 20.03
File Edit View Search Project Build Debug Fortran wxSmith Tools Tools+ Plugins Doxygen Settings Help
<global> main() int
Start here X Bai3.8.cpp X
6 10
7 1 2 3 4 5
-1+2+4+5=10
Process returned 0 (0x0) execution time : 5.087 s
Press any key to continue.
|
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
    sum = m_videu[top.y] * top.y; // Tính tổng trọng lượng
    s.push(state(top.x, top.y + 1)); // Thêm vào ngăn xếp
    ++top.y; // Tăng chỉ số trạng thái
    cout << "->" << endl; // Nhập khoảng cách cách nhau mỗi lần in -1
}
cout << "->" << endl; // Nhập khoảng cách cách nhau mỗi lần in -1
return 0; // Kết thúc chương trình
}

Logs & others
CodeBlocks X Search results X Ccc X Build log X CppCheck/Vera++ X CppCheck/Vera++ messages X Cscope X Debugger X Doxygen X Fortran info X Closed files list X Thread search X
File Line Message
== Build file: "no targets" in "no project" (compiler: unknown) ==

C:\Users\DELL\Desktop\KTLBuoi3\Bai3.8.cpp

21C C#
Windows (CR+LF) UTF-8 Line 17, Col 69, Pos 809 Insert Read/Write default ENG US SOS PM 12/6/2024

```

Hình 3.8.3 Test mẫu bài 8

Bài 3.9. Một y tá cần lập lịch làm việc trong N ngày, mỗi ngày chỉ có thể là làm việc hay nghỉ ngơi. Một lịch làm việc là tốt nếu không có hai ngày nghỉ nào liên tiếp và mọi chuỗi ngày tối đa làm việc liên tiếp đều có số ngày thuộc đoạn $[K1, K2]$. Hãy liệt kê tất cả các cách lập lịch tốt, với mỗi lịch in ra trên một dòng một xâu nhị phân độ dài n với bit 0/1 tương ứng là nghỉ/làm việc. Các xâu phải được in ra theo thứ tự từ điển

Dữ liệu vào:

Ghi 3 số nguyên $N, K1, K2$ ($N \leq 200, K1 < K2 \leq 70$)

Kết quả:

Ghi danh sách các lịch tìm được theo thứ tự từ điển

```

1 // Bai11.cpp - Created 13/10/2023 -744469 -20241 -
2
3 #include <iostream.h>
4 using namespace std;
5 const int MAX_viduc = 1000;
6 const int MOD_viduc = 1000000000 + 7;
7 int n_viduc, k1_viduc, k2_viduc;
8 int x[MAX_viduc];
9 int sol_viduc = 0;
10
11 // Ham nhap du lieu
12 void inputData(){
13     cin >> n_viduc >> k1_viduc >> k2_viduc;
14 }
15
16
17 // Ham kiem tra diai thien cho vien dau gia tri cho bit doi va ket a_viduc
18 bool check(int a_viduc, int i){
19     if(a_viduc == 0) return true;
20     else {
21         if(i == 0){
22             // Neu bit xong da la 0, khong co 2 next moi lien tiep
23             if(x[a_viduc - 1] == 0) return false;
24         }
25         else {
26             // Neu bit xong da la 1, cua dia bac ai moay lam vien lien tiep khong voat qua k1_viduc
27             if(sol_viduc < k1_viduc) return false;
28         }
29     }
30 }
31
32 if(x[a_viduc - 1] == 0){
33     // Neu bit xong da la 0, cua dia bac ai moay lam vienlien tiep ta a_viduc dia a_viduc khong la hon k1_viduc
34     if(n_viduc - a_viduc + 1 < k1_viduc) return false;
35 }
36
37 // Neu bit xong da la 1, cua dia bac ai moay lam vienlien tiep khong voat qua k2_viduc
38 if(sol_viduc >= k2_viduc) return false;
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60

```

Log & others

```

File Line Message
--- -----
== Build file: "no target" in "no project" (compiler: unknown)
== Build finished: 0 error(s), 0 warning(s) (0 minute(s), 1 second(s))

```

Hình 3.9.1 Code bài 9

```

25
26     else {
27         // Neu bit xong da la 1, cua dia bac ai moay lam vienlien tiep khong voat qua k1_viduc
28         if(sol_viduc < k1_viduc) return false;
29     }
30 }
31
32 else {
33     if(x[a_viduc - 1] == 0){
34         // Neu bit xong da la 0, cua dia bac ai moay lam vienlien tiep ta a_viduc dia a_viduc khong la hon k1_viduc
35         if(n_viduc - a_viduc + 1 < k1_viduc) return false;
36     }
37     else {
38         // Neu bit xong da la 1, cua dia bac ai moay lam vienlien tiep khong voat qua k2_viduc
39         if(sol_viduc >= k2_viduc) return false;
40     }
41 }
42
43 // Ham in ra man hinh lam vien lac
44 void solution(){
45     for(int i = 0; i <= n_viduc; i++){
46         cout << x[i];
47     }
48 }
49
50 // Ham thong bao cua truong ham
51 void TRY(int a_viduc){
52     for(int i = 0; i <= n_viduc; i++){
53         if(check(a_viduc, i)){
54             n_viduc = i;
55             int pre = sol_viduc;
56             if(i == 1){
57                 if(x[a_viduc - 1] == 1) sol_viduc++;
58                 else sol_viduc--;
59             }
60         }
61     }
62 }
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100

```

Log & others

```

File Line Message
--- -----
== Build file: "no target" in "no project" (compiler: unknown)
== Build finished: 0 error(s), 0 warning(s) (0 minute(s), 1 second(s))

```

```

10
11
12
13
14 // Häm in ra kết quả làm việc của
15 void solution()
16 {
17     for(int i = 0; i <= n_vidauc; i++)
18     {
19         cout << x[i];
20         cout << endl;
21     }
22 }
23
24 // Häm thử tất cả các trường hợp
25 void TRY(int a_vidauc)
26 {
27     for(int i = 0; i <= l2; i++)
28     {
29         if(check(a_vidauc, i))
30         {
31             x[a_vidauc] = i;
32             int pre = sol_vidauc;
33             if(i == 1)
34             {
35                 // Nếu bit này là 1, số phần tử mảng làm việc liên tiếp
36                 // bằng số bit - 1 ==> sol_vidauc++;
37                 sol_vidauc++;
38                 else sol_vidauc = 1;
39             }
40             else
41             {
42                 // Nếu bit này là 0, không có phần tử làm việc liên tiếp sau
43                 sol_vidauc = 0;
44             }
45             if(x[a_vidauc] == a_vidauc) solution(); // Nếu đã tìm thấy cho tất cả các bit, in ra kết quả
46             else TRY(a_vidauc + 1); // Nâng cấp: thử tất cả các bit tiếp theo
47             sol_vidauc = pre; // Phục hồi trạng thái mảng làm việc liên tiếp sau khi thử một giá trị
48         }
49     }
50 }
51
52 int main()
53 {
54     inputData();
55     TRY(1); // Đầu tiên thử với bit đầu tiên
56     return 0;
57 }

```

Logs & others

Code::Blocks X Search results X Ccc X Build log X Build messages X CppCheck/Vera++ X CppCheck/Vera++ messages X Scope X Debugger X Doxygen X Fortran info X Closed files list X Thread search X

Snipping Tool

Screenshot copied to clipboard
Automatically saved to screenshots folder.

Hình 3.9.3 Code bài 9(Tiếp)

Testcase 1:

```

10
11
12
13
14 // Häm in ra kết quả làm việc của
15 void solution()
16 {
17     for(int i = 0; i <= n_vidauc; i++)
18     {
19         cout << x[i];
20         cout << endl;
21     }
22 }
23
24 // Häm thử tất cả các trường hợp
25 void TRY(int a_vidauc)
26 {
27     for(int i = 0; i <= l2; i++)
28     {
29         if(check(a_vidauc, i))
30         {
31             x[a_vidauc] = i;
32             int l = sol_vidauc;
33             if(l)
34             {
35                 if(x[a_vidauc] == a_vidauc) Process returned 0 (0x0) execution time : 2.376 s
36                 cout << endl;
37             }
38             else
39             {
40                 if(x[a_vidauc] == a_vidauc) cout << endl;
41                 else sol_vidauc++;
42             }
43         }
44     }
45 }
46
47 int main()
48 {
49     inputData();
50     TRY(1); // Đầu tiên thử với bit đầu tiên
51     return 0;
52 }

```

Logs & others

Code::Blocks X Search results X Ccc X Build log X Build messages X CppCheck/Vera++ X CppCheck/Vera++ messages X Scope X Debugger X Doxygen X Fortran info X Closed files list X Thread search X

Hình 3.9.4 Testcase 1 bài 9

Testcase 2:

A screenshot of the Code::Blocks IDE interface. The main window shows a C++ code editor with several files listed in the tabs at the top: BaI2.12.cpp, BaI2.8.cpp, BaI2.5.cpp, BaI2.10.cpp, and BaI2.11.cpp. The current file is BaI2.12.cpp. The code contains a main function and a TR() function. The TR() function prints binary strings to the console. The terminal window at the bottom shows the execution of the program, displaying the binary strings and the message "Process returned 0 (0x0) execution time : 0.769 s press any key to continue.". Below the terminal is a log window titled "Logs & others" which displays build messages.

Hình 3.9.5 Testcase 2 bài 9

Testcase 3:

A screenshot of the Code::Blocks IDE interface. The main window shows a C++ code editor with several files listed in the tabs at the top: C:\Users\DELL\Desktop\KLT\T1\T1.cpp, C:\Users\DELL\Desktop\KLT\T1\T2.cpp, C:\Users\DELL\Desktop\KLT\T1\T3.cpp, C:\Users\DELL\Desktop\KLT\T1\T4.cpp, and C:\Users\DELL\Desktop\KLT\T1\T5.cpp. The current file is C:\Users\DELL\Desktop\KLT\T1\T1.cpp. The code contains a main function and several helper functions. The terminal window at the bottom shows the execution of the program, displaying a large amount of binary output from these functions and the message "Process returned 0 (0x0) execution time : 0.932 s Press any key to continue.". Below the terminal is a log window titled "Logs & others" which displays build messages.

Hình 3.9.6 Testcase 3 bài 9

Bài 3.10 Khoảng cách Hamming giữa hai xâu cùng độ dài là số vị trí mà ký tự tại vị trí đó là khác nhau trên hai xâu. Cho S là xâu gồm n ký tự 0. Hãy liệt kê tất cả các xâu nhị phân độ dài n , có khoảng cách Hamming với S bằng H . Các xâu phải được liệt kê theo thứ tự từ điển

Dữ liệu vào:

Dòng đầu chứa T là số testcase

T dòng tiếp theo, mỗi dòng mô tả một testcase, ghi N và H ($1 \leq H \leq N \leq 16$)

Kết quả:

Với mỗi testcase, in ra danh sách các xâu thỏa mãn. In ra một dòng trống giữa hai testcase

Ví dụ:

Dữ liệu mẫu:

2
4 2
1 0

Kết quả mẫu:

0011
0101
0110
1001
1010
1100
0

```

1 // Bài 10.cpp - Page 13
2 // Viết chương trình -744469 ->0241 /*
3 #include<iostream>
4 using namespace std;
5 const int MAX = 1000;
6 int S_viduc[MAX]; // Khai báo mảng lưu trữ các video và N_video cho từng testcase
7 int n_viduc; // Khai báo biến n_video số lượng video
8 int x_viduc[20]; // Khai báo mảng lưu trữ giá trị của các video
9 int L_viduc; // Khai báo biến L_video số lượng video cần kiểm tra
10
11 // Hàm kiểm tra điều kiện để thực hiện kiểm tra lỗi Hamming
12 bool checkint(x_viduc, int i_viduc, int N_k_video, int R_k_video) {
13     if (L_viduc + 1) {
14         // Nếu L_video bằng 1 (dạng xét giá trị 0 trong giá trị chính)
15         if (x_viduc[0] == 0) return true; // Nếu giá trị chính là 0 thì không có lỗi Hamming
16         // Nếu L_video không bằng 1 (dạng xét giá trị 1 trong giá trị chính)
17         if (L_viduc + 1 < N_k_video) return false; // Nếu giá trị chính là 1 và L_video + 1 nhỏ hơn N_k_video (không cách Hamming là lỗi)
18     }
19     else {
20         // Nếu L_video không bằng 1 (dạng xét giá trị 0 trong giá trị chính),
21         // TA N_k_video = L_video (để không ảnh hưởng đến giá trị chính)
22         if (N_k_video == L_video) return true; // Nếu giá trị chính là 0 và L_video = N_k_video (không cách Hamming là lỗi)
23         // Nếu L_video không bằng 1 (dạng xét giá trị 1 trong giá trị chính)
24         if (N_k_video - L_video < L_video) return false; // Nếu giá trị chính là 1 và N_k_video - L_video < L_video (không cách Hamming là lỗi)
25     }
26 }
27
28 // Hàm tách giá trị chính và giá trị khác
29 void solution(int N_k_video) {
30     for (int i_viduc = 1; i_viduc <= N_k_video; i_viduc++) {
31         cout << x_viduc[i_viduc];
32     }
33 }
34
35 // Hàm thực hiện thuật toán quay lui
36
37 // Log & others
38 // Code-Blocks X Search results X Ccc X Build log X Build messages X CppCheck/Vera++ messages X Cscope X Debugger X Doxygen X Fortran info X Closed files list X Thread search X
39 File Line Message
40 === Build file: "no targets" in "no project" (compiler: unknown) ===
41 === Build finished: 0 errors(s), 0 warning(s) (0 minute(s), 0 second(s)) ===

```

Hình 3.10.1 Code bài 10

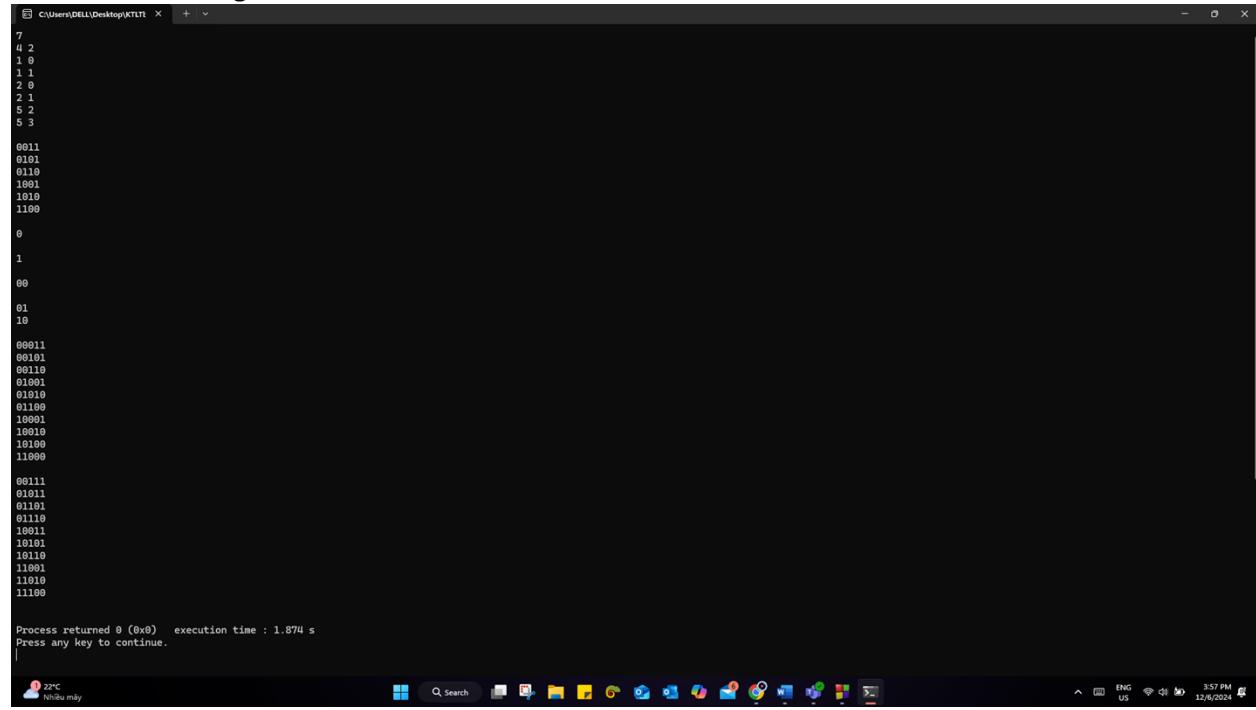
```

37 // Hàm tách giá trị chính và giá trị khác
38 void TRY(int x_video, int N_k_video, int H_k_video) {
39     for (int i_video = 0; i_video <= H_k_video;) {
40         if (checkint(x_video, i_video, N_k_video, H_k_video)) { // Kiem tra co kien tra xem co theu hieu lai cua video do khong
41             x_video[i_video] = 1; // Kiem tra co kien tra xem co theu hieu lai cua video do khong
42             if (i_video == H_k_video) { // Kiem tra co theu hieu lai cua video do khong
43                 if (x_video == H_k_video) solution(N_k_video); // Kien tra co theu hieu lai cua video do khong
44                 else {
45                     TRY(x_video + 1, N_k_video, H_k_video); // Kien tra co theu hieu lai cua video do khong
46                 }
47             }
48         }
49     }
50 }
51
52 // Hàm chính
53 int main() {
54     cin >> n_viduc; // Nhập số lượng video
55     for (int i_video = 0; i_video < n_viduc; i_video++) {
56         cin >> x_video[i_video]; // Nhập giá trị x của video và N_video cho từng testcase
57     }
58     cout << endl;
59
60 // Đặt giá trị testcases để thực hiện thuật toán
61 for (int i_video = 0; i_video < n_viduc; i_video++) {
62     TRY(i, S_viduc[i_video], H_k_video[i_video]); // Gọi hàm TRY với tham số là i_video -> x_video[i_video], H_k_video[i_video]
63     cout << endl; // In ra dòng trống giữa hai testcase
64 }
65
66
67 // Log & others
68 // Code-Blocks X Search results X Ccc X Build log X Build messages X CppCheck/Vera++ messages X Cscope X Debugger X Doxygen X Fortran info X Closed files list X Thread search X
69 File Line Message
70 === Build file: "no targets" in "no project" (compiler: unknown) ===
71 === Build finished: 0 errors(s), 0 warning(s) (0 minute(s), 0 second(s)) ===

```

Hình 3.10.2 Code bài 10(Tiếp)

Testcase 1:



```

7
4 2
1 0
1 1
2 0
2 1
5 2
5 3

0011
0101
0110
1001
1010
1100

0
1
00
01
10

00011
00101
00110
01001
01101
01100
10001
10010
10100
11000

00111
01011
01101
01110
10011
10101
10110
11001
11010
11100

Process returned 0 (0x0) execution time : 1.874 s
Press any key to continue.

```

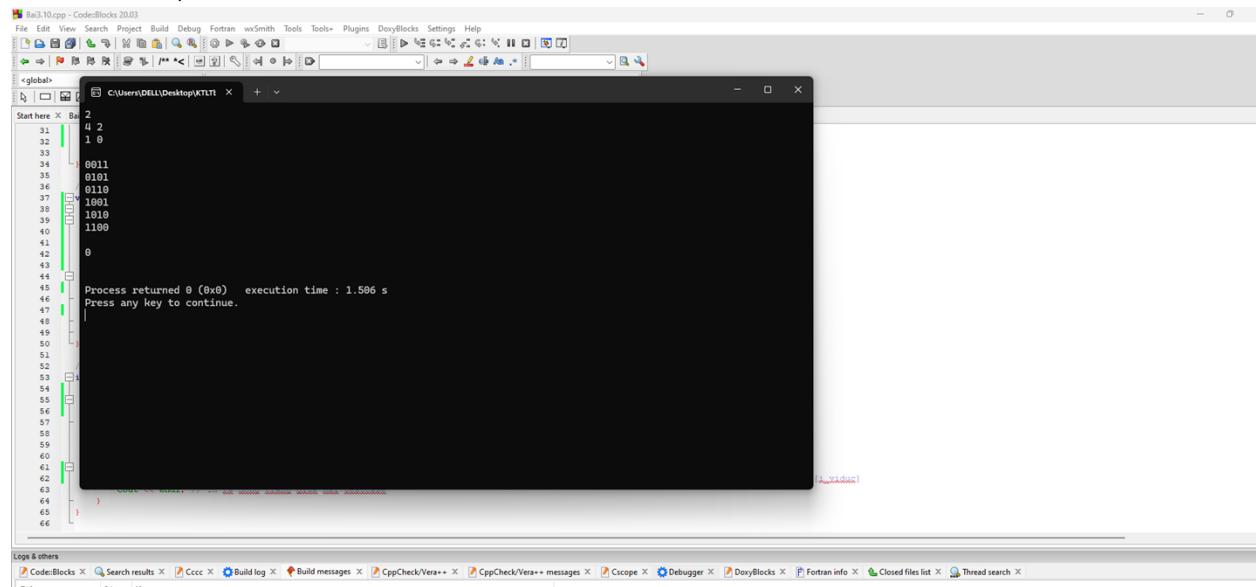
22°C
Nhieu mdy

Search | File | Edit | View | Search | Project | Build | Debug | Fortran | wxSmith | Tools | Tools+ | Plugins | DoxygenBlocks | Settings | Help

3:57 PM 12/6/2024

Hình 3.10.3 Testcase 1 bài 10

Testcase ví dụ;



```

2
4 2
1 0
0011
0101
0110
1001
1010
1100

0
1
00
01
10

00011
00101
00110
01001
01101
01100
10001
10010
10100
11000

00111
01011
01101
01110
10011
10101
10110
11001
11010
11100

Process returned 0 (0x0) execution time : 1.586 s
Press any key to continue.

```

File | Edit | View | Search | Project | Build | Debug | Fortran | wxSmith | Tools | Tools+ | Plugins | DoxygenBlocks | Settings | Help

Logs & others

Hình 3.10.4 Testcase ví dụ bài 10

Bài 3.11 Superior là một hòn đảo tuyệt đẹp với n địa điểm chụp ảnh và các đường một chiều nối các điểm chụp ảnh với nhau. Đoàn khách tham quan có r người với sở thích chụp ảnh khác nhau. Theo đó, mỗi người sẽ đưa ra danh sách các địa điểm mà họ muốn chụp. Bạn cần giúp mỗi người trong đoàn lập lịch di chuyển sao cho đi qua các điểm họ yêu cầu đúng một lần, không đi qua điểm nào khác, bắt đầu tại điểm đầu tiên và kết thúc tại điểm cuối cùng trong danh sách mà họ đưa ra, và có tổng khoảng cách đi lại là nhỏ nhất.

Dữ liệu vào:

Dòng đầu chứa n và r

Tiếp theo là ma trận $n \times n$ mô tả chi phí đi lại giữa các địa điểm. Chi phí bằng 0 có nghĩa là không thể đi lại giữa hai địa điểm đó.

r dòng tiếp theo chứa danh sách các địa điểm mà người r đưa ra. Lưu ý là hành mỗi hành trình cần phải bắt đầu và kết thúc bởi hai đỉnh đầu và cuối của danh sách, còn các địa điểm còn lại có thể thăm theo bất kỳ thứ tự nào.

Kết quả:

Gồm r dòng ghi chi phí đi lại ít nhất của r người theo thứ tự đầu vào.

Ví dụ:

Dữ liệu mẫu:

```
6 3
0 1 2 0 1 1
1 0 1 1 1 0
0 2 0 1 3 0
4 3 1 0 0 0
0 0 1 1 0 0
1 0 0 0 0 0
1 3 5
6 3 2 5
6 1 2 3 4 5
```

Kết quả mẫu:

```
5
0
7
```

```

1 // Ba3.11.cpp - CodeBlocks 20.03
2 File Edit View Search Project Build Debug Fortran wxSmith Tools Tools+ Plugins Doxygen Settings Help
3 Start here X Ba3.11.cpp X Ba3.11.cpp X Ba3.11.cpp X
4
5 // Ba3.11.cpp - ID: 13 - 744469 - 20241 - /
6 // Ba3.11.cpp - 2024-03-13 15:04:09 - 744469 - 20241 - /
7 //include<iostream>
8 using namespace std;
9 const int MAX_viduc = 100;
10 int r_viduc;
11 int price_viduc[MAX_viduc];
12 int x_viduc[MAX_viduc]; // Biên lục các lich trình đi chuyến xe mỗi ngày
13 bool visited_viduc[MAX_viduc]; // Biên lục tra xem một điểm có được thăm chưa
14 vector<int> vt_viduc; // Biên lục các điểm mà xe bus di chuyển hàng ngày
15 int best_viduc; // Biên lục giá trị tổng chi phí tối đa
16 int sum_price_viduc; // Biên lục tổng chi phí di lại của một lich trình
17 int start_viduc, des_viduc, numofPoint_viduc; // Biên lục từ điểm xuất phát, điểm đến, và số điểm cần di qua
18
19 // Biên lục giá trị
20 void input(){
21     cin >> n_viduc;
22     for(int i = 0; i < n_viduc; i++){
23         for(int j = 0; j < n_viduc; j++){
24             cin >> price_viduc[i][j]; // ma trận giá trị
25         }
26     }
27 }
28
29 // Biên lục tra xem một điểm có thể chay di di vien không
30 bool check(int a, int b){
31     if(visited_viduc[vt_viduc[a]]) return false; // Nếu đã thăm điểm đó rồi thì trả về false
32     if(price_viduc[x_viduc[a-1]][vt_viduc[b]] == 0) return false; // Nếu giá trị của điểm A là 0 thì trả về false
33     return true;
34 }
35
36 // Biên lục giá trị
37 void solution(){
38     if(price_viduc[x_viduc[numofPoint_viduc-1]][des_viduc] == 0) return; // Nếu giá trị chi phí cuối cùng là 0 thì không tính
39     best_viduc = min(best_viduc, sum_price_viduc + price_viduc[x_viduc[numofPoint_viduc-1]][des_viduc]); // Cập nhật giá trị chi phí mới nhất
40 }
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69

```

Logs & others

CodeBlocks X Search results X Ccc X Build log X Build messages X CppCheck/Vera++ X CppCheck/Vera++ messages X Cscope X Debugger X Doxygen X Fortan info X Closed files list X Thread search X

File Line Message

==> Build file: "no targets" in "no project" (compiler: unknown) ==>

==> Build finished: 0 errors(s), 0 warning(s) (0 minute(s), 1 second(s)) ==>

Hình 3.11.1 Code bài 11

```

1 // Ba3.11.cpp - CodeBlocks 20.03
2 File Edit View Search Project Build Debug Fortran wxSmith Tools Tools+ Plugins Doxygen Settings Help
3 Start here X Ba3.11.cpp X Ba3.11.cpp X Ba3.11.cpp X
4
5 // Ba3.11.cpp - ID: 13 - 744469 - 20241 - /
6 // Ba3.11.cpp - 2024-03-13 15:04:09 - 744469 - 20241 - /
7 //include<iostream>
8 using namespace std;
9 const int MAX_viduc = 100;
10 int r_viduc;
11 int price_viduc[MAX_viduc];
12 int x_viduc[MAX_viduc]; // Biên lục các lich trình
13 bool visited_viduc[MAX_viduc]; // Biên lục giá trị chi phí cuối cùng là 0 khi không tính
14 vector<int> vt_viduc; // Biên lục các điểm mà xe bus di chuyển
15 int best_viduc = min(best_viduc, sum_price_viduc + price_viduc[x_viduc[numofPoint_viduc-1]][des_viduc]); // Cập nhật giá trị chi phí mới nhất
16
17 // Biên lục giá trị
18 void solution(){
19     if(price_viduc[x_viduc[numofPoint_viduc-1]][des_viduc] == 0) return; // Nếu giá trị chi phí cuối cùng là 0 thì không tính
20     best_viduc = min(best_viduc, sum_price_viduc + price_viduc[x_viduc[numofPoint_viduc-1]][des_viduc]); // Cập nhật giá trị chi phí mới nhất
21 }
22
23 // Biên lục giá trị
24 void TRY(int a){
25     for(int i=1; i<numofPoint_viduc-1; i++){
26         if(price_viduc[a][i] > 0){
27             visited_viduc[vt_viduc[i]] = true; // Biên lục đã thăm điểm i
28             sum_price_viduc += price_viduc[x_viduc[a-1]][vt_viduc[i]]; // Cập nhật giá trị tổng chi phí (tính cả điểm i)
29             x_viduc[a] = vt_viduc[i]; // Cập nhật lich trình
30             if(solution()) // Nếu đã qua tất cả các điểm, thì xem lich trình có hợp lệ không
31                 else
32                     TRY(a+1); // Nếu chưa, thử另行 một điểm để tái
33             visited_viduc[vt_viduc[i]] = false; // Nếu lich trình thất bại, trả về giá trị cũ
34             sum_price_viduc -= price_viduc[x_viduc[a-1]][vt_viduc[i]] // Giảm chi phí di lại
35         }
36     }
37 }
38
39 int main(){
40     input(); // Nhập dữ liệu
41
42     // Biển lục giá trị
43     while(r_viduc > 0){
44         best_viduc = INT_MAX;
45         sum_price_viduc = 0;
46
47         // Tạo danh sách các điểm không thuộc chuỗi lich trình
48         int location;
49         while (cin >> location){
50             vt_viduc.push_back(location - 1);
51         }
52
53         TRY(0); // Khởi tạo, thử另行 điểm đầu tiên
54     }
55 }
56
57
58
59
60
61
62
63
64
65
66
67
68
69

```

Logs & others

CodeBlocks X Search results X Ccc X Build log X Build messages X CppCheck/Vera++ X CppCheck/Vera++ messages X Cscope X Debugger X Doxygen X Fortan info X Closed files list X Thread search X

File Line Message

==> Build file: "no targets" in "no project" (compiler: unknown) ==>

==> Build finished: 0 errors(s), 0 warning(s) (0 minute(s), 1 second(s)) ==>

Hình 3.11.2 Code bài 11 (Tiếp)

```

49     else TRY(a+1); // Khi chưa, thử nút kia điểm số kia
50
51     visited_viduc[vt_viduc[i]] = false; // Khi lai trong thoi visited_viduc di thi nút kia lúu chon khác
52     sum_price_viduc -= price_viduc[x_viduc[a-1]][vt_viduc[i]]; // Giảm giá trị di lai
53
54   }
55
56 }
57
58 int main()
59 {
60   input(); // Nhập dữ liệu
61
62   // Đặt que tăng dần từ trái
63   while(r_viduc > 0)
64   {
65     best_viduc = INT_MAX;
66     sum_price_viduc = 0;
67
68     // Tính danh sách các địa điểm tăng dần mỗi lần chon
69     int location;
70     while(cin >> location)
71     {
72       vt_viduc.push_back(location - 1);
73
74       if(cin.get() == '\n') break;
75     }
76
77     // Khi que đã đầy sẽ bắt đầu quay lại
78     start_viduc = vt_viduc[vt_viduc.size() - 1];
79     des_viduc = vt_viduc[vt_viduc.size() - 1];
80     numOfPoint = vt_viduc.size();
81
82     x_viduc[0] = start_viduc; x_viduc[numOfPoint - 1] = des_viduc;
83     for(int i = 0; i < viduc; i++)
84     {
85       visited_viduc[i] = false;
86     }
87
88     TRY(1); // Thoát khỏi quay lại để tìm lối trình
89   }
90
91   cout << endl;
92
93   return 0;
94 }

```

Logs & others

Code::Blocks X Search results X Ccc X Build log X Build messages X CppCheck/Vera++ messages X Cscope X Debugger X Doxygen X Fortan info X Closed files list X Thread search X

File Line Message

==> Build file: "no targets" in "no project" (compiler: unknown) ==>

==> Build finished: 0 errors(s), 0 warning(s), 0 minute(s), 1 second(s) ==>

Hình 3.11.3 Code bài 11 (Tiếp)

Testcase 1:

```

36   best_viduc = min(best_viduc, sum_price_viduc + price,
37   - );
38
39   // Ham quay lui de thoi nhanh cac lich trinh
40   void TRY(int a)
41   {
42     for(int i = 1; i < numOfPoint - 1; i++)
43     {
44       if(visited[i] == true)
45       {
46         visited_viduc[vt[i]] = true; // danh dau
47         sum_price_viduc += price_viduc[x[a-1]][vt[i]];
48
49         x[a] = vt[i]; // sau nhanh cac lich trinh
50         if(i == numOfPoint - 1)
51           solution(); // Khi đã đc que kia và kết
52         else
53           TRY(a+1); // Khi chưa, thử nút kia
54       }
55     }
56
57 }
58
59 int main()
60 {
61   freopen("input.txt", "r", stdin);
62   freopen("output.txt", "w", stdout);
63   input(); // Nhập dữ liệu
64
65   // Đặt que tăng dần từ trái
66   while(r > 0)
67   {
68     best_viduc = INT_MAX;
69     sum_price_viduc = 0;
70
71     // Tính danh sách các địa điểm tăng dần mỗi lần
72     int location;
73     while(cin >> location)
74     {
75       vt.push_back(location - 1);
76     }
77
78     TRY(1); // Thoát khỏi quay lại để tìm lối trình
79   }
80
81   cout << endl;
82
83   return 0;
84 }

```

Logs & others

Code::Blocks X Search results X Ccc X Build log X Build messages X CppCheck/Vera++ messages X Cscope X Debugger X Doxygen X Fortan info X Closed files list X Thread search X

File Line Message

==> Build file: "no targets" in "no project" (compiler: unknown) ==>

==> Build finished: 0 error(s), 0 warning(s), 0 minute(s), 1 second(s) ==>

Hình 3.11.4 Test case 1 bài 11

Testcase 2:

```

36     best_viduc = min(best_viduc, sum_price_viduc + price);
37
38 // Khi chưa có thành phần nào của tách kinh
39
40 void TKT(int a){
41     for(int i=1; i<numOfPoint-1; i++){
42         if(check(a, i))
43             visited_viduc[vt[i]] = true; // đánh dấu đã
44             sum_price_viduc += price_viduc[x[i-1]][vt[i]];
45
46         x[a] = vt[i]; // xác nhận tách k thành
47         if(a == numOfPoint-2)
48             solution[0] // kia là số thứ tự
49         else
50             TKT(a+1); // kia chia thành phần nào đó
51
52         visited_viduc[vt[i]] = false; // xác minh xem
53         sum_price_viduc += price_viduc[x[i-1]][vt[i]];
54     }
55 }
56
57
58 int main(){
59     freopen("input.txt", "r", stdin);
60     freopen("output.txt", "w", stdout);
61     input();
62
63 // Banh que xanh nong xanh daon
64 while(c > 0){
65     best_viduc = INT_MAX;
66     sum_price_viduc = 0;
67
68     // Ban daon sach cac dia diem nong nong manh oh
69     int location;
70     while(cin >> location){
71         vt.push_back(location - 1);
    }
}

Logs & others
File Line Message
C:\Users\DELL\Desktop\XTL\Buu\bau11.cpp
12
25
39
40
27
25
6

```

Hình 3.11.5 Test case 2 bài 11

Bài 3.12 Cho đồ thị vô hướng G , hãy đếm số đường đi đi qua k cạnh và không đi qua đỉnh nào quá một lần.

Dữ liệu vào:

Dòng 1: Chứa hai số nguyên n và k ($1 \leq n \leq 30, 1 \leq k \leq 10$) với n là số đỉnh của G . Các đỉnh sẽ được đánh số từ 1 đến n

Dòng 2: Chứa số nguyên m ($1 \leq m \leq 60$) là số cạnh của G ;

m dòng tiếp theo: Mỗi dòng chứa hai số nguyên là một cạnh của G

Kết quả:

Số lượng đường đi đơn độ dài k

Ví dụ:

Dữ liệu mẫu:

4 3

5

1 2

1 3

1 4

2 3

3 4

Kết quả mẫu:

6

```

1 // Bai12.cpp - Code::Blocks 20.03
2 // Vi Hung Duc - 4836 - 744469 - 20241 *
3 #include <iostream>
4 using namespace std;
5 const int MAX = 100;
6 int n_viduc, k_viduc;
7 int m_viduc;
8 vector<int> > vt; // Chuyển từ file .txt các số sang file .txt danh sách k
9 int x_viduc[MAX];
10 bool visited[viduc][MAX];
11 int res_viduc;
12
13 void input(){
14     cin >> n_viduc >> k_viduc; // Nhập số k
15     cin >> m_viduc;
16     vt.resize(n_viduc); // Resize vector vt để có thể lưu trữ danh sách k các số định (từ 0 đến n_viduc)
17     for(int i=0; i<m_viduc; i++){
18         int tmp1, tmp2;
19         cin >> tmp1 >> tmp2;
20
21         vt[tmp1-1].push_back(tmp2-1); // Dịch tmp1 và tmp2 sang
22         vt[tmp2-1].push_back(tmp1-1); // Dịch tmp2 và tmp1 sang
23     }
24
25     for(int i=0; i<n_viduc; i++){
26         visited[viduc][i] = false; // Đặt giá trị ban đầu là chưa thăm
27     }
28     res_viduc = 0; // Khởi tạo biến res_viduc để đếm số lượng thăm dò
29 }
30
31 bool check(int a, int b){ // Kiểm tra
32     if(a == 0) return true; // Nếu a = 0, tức là danh sách rỗng, luôn là một đường đi hợp lệ
33     if(visited[viduc][b]) return false; // Nếu đã thăm dò & kia kia là false
34 }
35
36 int index = 0;
37 for(int j=0; j<vt[x_viduc(a-1)].size(); j++){
38
39

```

Logs & others

File	Line	Message
C:\Users\DELL\Desktop\KTLBuoi3\Bai12.cpp	31	** Build file: "no targets" in "no project" (compiler: unknown) ** ** Build finished: 0 error(s), 0 warning(s) (0 minute(s), 1 second(s)) **

Hình 3.12.1 Code bài 12

```

32     if(a == 0) return true; // Nếu a = 0, tức là đỉnh xuất phát luôn là một đỉnh có ham số
33     if(visited_viduc[1]) return false; // Nếu đã thăm đỉnh i thì trả về false
34
35     int index = 0;
36     for(int j=0; j<vt[x_viduc[a-1]].size(); j++){
37         if(i == vt[x_viduc[a-1][j]]) index++; // Kiểm tra xem i có là đỉnh xuất của không
38     }
39     if(index == 0) return false; // Nếu không có, trả về false
40
41     return true; // Không sao, trả về true
42 }
43
44 void solution(){
45     res_viduc++; // Khi tìm thấy một đường đi hợp lệ, tăng biến đếm res_viduc lên 1
46 }
47
48 void TRY(int a){
49     for(int i=0; i<n_viduc; i++){
50         if(visited_viduc[i] == true)
51             visited_viduc[i] = false;
52         x_viduc[a] = i;
53
54         if(a == k_viduc) solution(); // Nếu a là đỉnh xuất, gọi hàm solution để tăng biến đếm
55         else TRY(a+1); // Kế đến, tìm đỉnh sau của nó
56
57         visited_viduc[i] = true; // Đánh dấu đỉnh i là được thăm
58     }
59 }
60
61 int main(){
62     input(); // Nhập dữ liệu
63     TRY(0); // Gọi hàm TRY để bắt đầu thử nghiệm các đường đi
64     cout << res_viduc / 2; // Số đường đi là res_viduc chia 2 vì mỗi đường đi đều đếm hai lần (u -> v và v -> u)
65 }
66

```

Logs & others

Code:Blocks X Search results X Cccc X Build log X Build messages X CppCheck/Vera++ messages X Cscope X Debugger X DosyBlocks X Fortran info X Closed files list X Thread search X

File Line Message

File Line Message

C:\Users\DELL\Desktop\KTL\Ba12.cpp

21C Nhuhi mly

Hình 3.12.2 Code bài 12 (Tiếp)

Testcase 1:

```

1 #include<iostream>
2 #include<vector>
3 using namespace std;
4 const int n = 10;
5 const int m = 20;
6 int n_viduc = 6;
7 int k_viduc = 4;
8 vector<vector<int>> vt;
9 int x_viduc[10];
10 bool visited[10];
11 int res_viduc = 0;
12
13 void input()
14 {
15     cin >> n >> m;
16     vt.resize(n);
17     for(int i=0; i<n; i++)
18     {
19         vt[i].resize(m);
20         for(int j=0; j<m; j++)
21         {
22             cin >> vt[i][j];
23         }
24     }
25 }
26
27 void solve()
28 {
29     TRY(0);
30 }
31
32 bool check()
33 {
34     for(int i=0; i<n; i++)
35     {
36         int in = 0;
37         for(int j=0; j<m; j++)
38         {
39             if(vt[i][j] == 1)
40                 in++;
41         }
42         if(in != x_viduc[i])
43             return false;
44     }
45     return true;
46 }
47
48 void TRY(int a)
49 {
50     for(int i=0; i<n; i++)
51     {
52         if(visited[i] == true)
53             visited[i] = false;
54         x_viduc[a] = i;
55
56         if(a == k_viduc) solve();
57         else TRY(a+1);
58
59         visited[i] = true;
60     }
61 }
62
63 int main()
64 {
65     input();
66     solve();
67     if(check())
68     {
69         cout << res_viduc;
70     }
71 }

```

Logs & others

Code:Blocks X Search results X Cccc X Build log X Build messages X CppCheck/Vera++ messages X Cscope X Debugger X DosyBlocks X Fortran info X Closed files list X Thread search X

File Line Message

File Line Message

C:\Users\DELL\Desktop\KTL\Ba12.cpp

Testcase 2:

```

20 5
5b
18 3
8 7
8 2
15 7
12 3
18 5
13 4
10 6
17 13
6 5
3 19
6 17
13 8
9 6
10 2
8 9
15 19
1 10
1 16
1 3
4 1
12 10
14 1
5 20
15 17
1 1
1 3
2 1
14 15
1 1
4 15
8 12
14 1
19 16
19 6
18 11
1 5
4 5
12 1
4 2
13 11
14 11
7 1
2 14
1 18
10 1
13 4
3 4
3 4
11 5
9 19
16565
Process returned 0 (0x0) execution time : 0.834 s
Press any key to continue.

```

Hình 3.12.4 Testcase 2 bài 12

Testcase 3:

```

20 5
5b
18 3
8 7
8 2
15 7
12 3
18 5
13 4
10 6
17 13
6 5
3 19
6 17
13 8
9 6
10 2
8 9
15 19
1 10
1 16
1 3
4 1
12 10
14 1
5 20
15 17
1 1
1 3
2 1
14 15
1 1
4 15
8 12
14 1
19 16
19 6
18 11
1 5
4 5
12 1
4 2
13 11
14 11
7 1
2 14
1 18
10 1
13 4
3 4
3 4
11 5
9 19
16565
Process returned 0 (0x0) execution time : 0.886 s
Press any key to continue.

```

Hình 3.12.5 Testcase 3 bài 12

Testcase 4:

```

28 6
58 4
18 18
18 18
18 9
12 7
19 1
2 9
16 18
12 12
10 11
17 13
18 20
7 3
6 18
6 8
17 14
3 19
6 11
6 7
20 13
16 14
14 14
12 16
19 4
18 8
1 3
12 11
3 15
10 10
1 17
6 16
3 23
17 8
19 15
18 5
3 10
19 2
9 10
1 12
13 5
11 17
1 2
19 11
11 7
20 9
6 10
1 6
18 9
16 11
11 20
03781
Process returned 0 (0x0) execution time : 2.581 s
Press any key to continue.

```

Hình 3.12.6 Testcase 4 bài 12

Testcase 5:

```

28 8
59 2
17 28
15 13
11 6
9 10
9 10
17 1
5 13
18 28
15 10
2 6
6 3
17 18
3 19
13 2
7 16
8 7
5 20
7 15
9 11
15 5
2 12
12 16
6 1
14 8
18 13
1 4
4 19
9 1
2 19
1 16
1 14
2 18
16 5
6 1
19 18
9 11
15 17
14 17
1 9
15 3
3 16
8 1
3 2
13 14
18 11
6 19
1 1
20 1
5 12
12 28
075911
Process returned 0 (0x0) execution time : 1.384 s
Press any key to continue.

```

Hình 3.12.7 Testcase 5 bài 12

Testcase 6:

```

30 3
60
72
7 38
26 29
9 4
2 28
1 5
30 5
20 23
20 9
46
1 13
21 25
11 2
16 9
22 21
1 4
27 20
0 3
0 18
1 17
1 7
10 19
20 13
2 10
0 18
0 30
15 6
29 16
11 11
28 20
26 13
30 13
20 7
19 23
18 3
21 27
17 18
1 15
7 26
24 22
9 7
1 11
18 8
7 3
18 2
27 2
10 1
0 11
0 10
21 8
19 8
1 5
16 28
26 25
11 27
14 22
11 1
19 26
12 17
802
Process returned 0 (0x0) execution time : 0.899 s
Press any key to continue.

```

Hình 3.12.8 Testcase 6 bài 12

Testcase 7:

```

30 10
60
30 29
1 1
28 24
25 15
20 15
29 11
21 17
7 25
27 2
14 7
9 11
27 23
21 30
8 23
25 14
18 13
29 11
13 25
23 25
1 25
14 2
1 2
10 9
3 27
22 3
7 5
16 24
16 3
13 13
17 2
2 8
12 15
29 28
26 13
17 6
5 9
1 21
23 21
14 21
5 16
6 26
12 12
21 5
20 14
13 13
20 30
14 30
12 12
26 22
13 6
9 18
9 23
9 4
21 16
12 16
14 26
6 11
5 11
27 2
13 20
14 15
Process returned 0 (0x0) execution time : 2.981 s
Press any key to continue.

```

Hình 3.12.9 Testcase 7 bài 12

Testcase 8:

```

30 16
69
2 19
9 29
13 23
19 28
9 28
28 23
14 29
4 22
6 22
9 23
7 12
18 27
9 23
13 6
12 19
29 1
12 1
28 18
18 23
8 13
23 17
1 9
24 9
11 13
14 6
13 15
23 29
18 14
29 16
8 9
14 7
29 15
16 15
12 14
3 8
5 22
5 18
29 16
9 8
18 9
26 14
13 16
3 16
12 28
6 9
18 24
29 28
13 28
29 23
5 19
20 13
22 1
4 28
29 2
29 1
13 24
1 4
23 5
15 1
6 12
1865192
Process returned 0 (0x0) execution time : 5.282 s
Press any key to continue.

```

Hình 3.12.10 Testcase 8 bài 12

Testcase 9:

```

30 11
68
13 29
4 15
4 9
17 20
1 2
11 16
20 12
29 27
23 1
24 23
10 6
9 8
1 25
24 1
21 11
21 29
27 14
1 3
30 1
9 1
1 20
2 26
18 25
2 7
2 29
3 23
1 13
3 18
15 14
9 18
27 1
21 23
15 29
9 8
14 9
3 11
8 1
20 24
1 17
4 18
27 1
11 14
13 12
13 19
18 27
27 11
1 2
14 23
8 23
8 19
17 13
29 5
22 1
13 27
15 19
7 38
13 1
21 15
9 10
1865192
Process returned 0 (0x0) execution time : 7.437 s
Press any key to continue.

```

Hình 3.12.11 Testcase 9 bài 12