Bài 16:

*Thiết kế thuật toán giải bài toán **xếp tám hậu** theo phương pháp quay lui với các công việc sau:*

1. Nêu bài toán;

2. Mô tả chi tiết thuật toán;

3. Thực hiện từng bước việc đặt 5, 6, 7 con hậu đầu tiên trên bàn cờ 8x8;

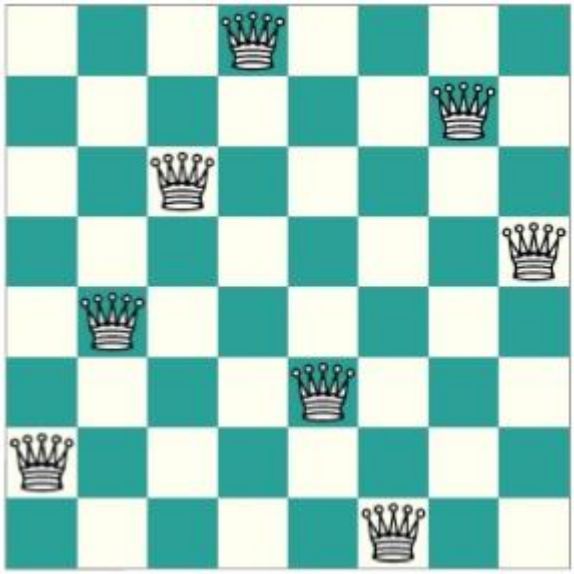
4. Viết chương trình sử dụng C, C++;

5. Đánh giá độ phức tạp thuật toán theo lý thuyết;

6. Viết báo cáo (trình bày các nội dung từ 1-5).

1. Nêu bài toán

Trên một bàn cờ vua 8×8, đặt tám quân hậu (không phân biệt màu sắc) sao cho không có quân hậu nào có thể "ăn" được quân hậu khác, hay nói cách khác, không có một cặp hai quân hậu nào cùng nằm trên một hàng, một cột hoặc một đường chéo.



2. Mô tả chi tiết thuật toán

a) Ý tưởng của phương pháp quay lui

- Ý tưởng chính của phương pháp quay lui là các bước hướng tới lời giải cuối cùng của bài toán dựa trên việc **Thử-và-Sai**.

• Tại mỗi bước:

- Nếu có 1 lựa chọn được chấp nhận thì ghi nhận lại lựa chọn này và và tiến hành các bước thử tiếp theo;
- Nếu tất cả các lựa chọn không được chấp nhận thì trở lại bước trước, xóa bỏ sự ghi nhận của ứng viên và chọn lựa ứng viên tiếp theo.

❖ **Chú ý:**

- Khi quay lui điểm quan trọng của thuật toán là phải ghi nhớ tại mỗi bước đi để tránh trùng lặp khi quay lui.
- Dễ thấy cấu trúc ngăn xếp khá phù hợp để lưu trữ các thông cần ghi nhớ như đề cập ở trên.
- Đệ qui là kỹ thuật thường được sử dụng trong phương pháp quay lui.

❖ **Lược đồ chung**

- Lời giải bài toán có thể mô tả dạng 1 vector n chiều $x = (x_1, x_2, .., x_n)$ thỏa mãn một điều kiện nào đó.
- Giả sử đã xây dựng được i-1 thành phần (x_1, x_2,.., x_{i-1}), cần xác định thành phần thứ i:
  - Nếu khả năng k nào đó phù hợp -> lấy x_i=k, ghi nhận trạng thái đã dùng của k. Nếu i=n -> có được 1 lời giải.
  - Nếu không có khả năng nào cho x_i thì quay lui và chọn lại x_{i-1}.

```
Try(i) ≡
  for ( j = 1 → k)
        If ( xi chấp nhận được khả năng j)
        {
              Xác định xi theo khả năng j;
              Ghi nhận trạng thái mới;
              if( i < n)
                    Try(i+1);
              else
                    Ghi nhận nghiệm;
              Trả lại trạng thái cũ cho bài toán;
        }
```

b) Ý tưởng thuật toán tám hậu

❖ Ý tưởng bài toán 8 hậu

1. Lần lượt xếp các con hậu vào bàn cờ
2. Giả sử đã xếp được i con hậu (từ 1 đến i)
3. Xếp hậu thứ i+1

a. Nếu tìm được 1 ô hợp lệ (không bị các con hậu trước đó ăn)

=> xếp hậu thứ i+1 vào vị trí vừa tìm thấy. Lặp lại bước 3.

b. Nếu không tìm được ô hợp lệ

=> tìm vị trí phù hợp khác để đặt lại hậu thứ i.

❖ Mấu chốt của thuật toán rõ ràng là xét xem có thể đặt quân hậu tiếp theo như thế nào. Theo luật cờ vua, một quân hậu có thể ăn các quẩn khác nếu nằm trên cùng 1 đường, đường này có thể là :

- Hàng
- Cột
- Các đường chéo (đi qua tọa độ vị trí của hậu).

=> Mỗi hàng chỉ có thể chứa 1 và chỉ 1 quân hậu. Nên việc chọn vị trí cho quân hậu thứ i có thể giới hạn được ở hàng thứ i. Như thế tham số i trở thành chỉ hàng, và quá trình chọn vị trí cho quân hậu tiến hành trên toàn giá trị có thể có của các cột j.

❖ Quy ước:

x[i]: Chỉ quân hậu thứ i: nằm ở hàng i

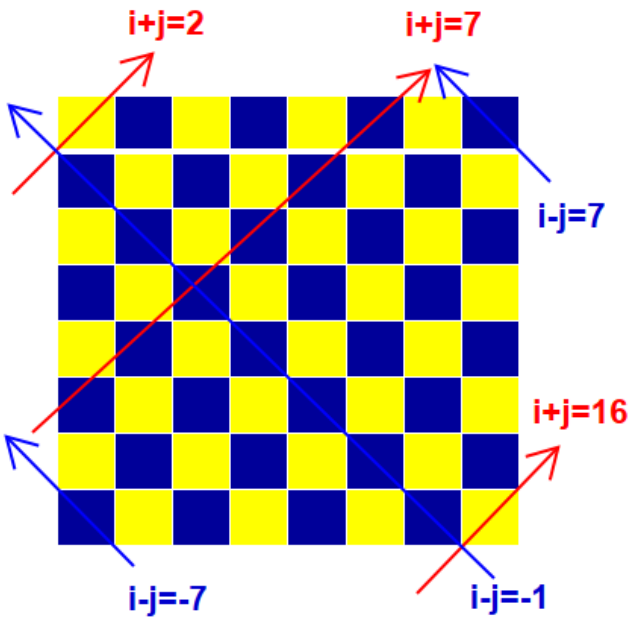
x[i] = j : Quân hậu thứ i đặt ở cột j (khi đó cần đánh dấu đã được chọn để bước sau không chọn lại)

- Để quân hậu i (trên hàng i) chấp nhận cột j thì cột j và 2 đường chéo qua ô <i, j> phải còn trống tức là không có quân hậu khác chiếm.

Lưu ý rằng trong 2 đường chéo :

- Đường chéo chính là đường chéo gồm các ô mà chỉ số dòng i = cột j
- Đường chéo thuận (vuông góc với đường chéo chính) gồm tất cả các ô (i, j) có (i+j) là hằng số
- Đường chéo nghịch (song song với đường chéo chính) gồm tất cả các ô (i, j) có (i-j) là hằng số

- Hậu ở dòng i, chỉ được đặt vào cột j nếu i-1 hậu đã đặt trước đó không "ăn" được hậu ở vị trí [i,j] (dòng i, cột j).
- Trên đường chéo đỏ:
  - Giá trị i+j là hằng số
  - Có giá trị từ 2 đến 16
- Trên đường chéo xanh:
  - Giá trị i-j là hằng số
  - Có giá trị từ -7 đến 7



a[j] = 0 : cột j đã bị chiếm

a[j] = 1 : không có quân hậu nào ở cột j.

b[i+j] = 1: không có quân hậu nào ở đường chéo thuận (i+j).

c[i-j] = 1 : không có quân hậu nào là đường chéo nghịch (i-j).

=> Hậu i (dòng i) được đặt vào cột j nếu:

b[i+j] = 1 và c[i-j] = 1

c) Biểu diễn thuật toán

Cài đặt

Khởi tạo

```
a[j]   = 1
b[i+j] = 1
c[i -j] = 1
```

```
Try(i) ≡
{
   for (j = 1; j <= 8; j++)
      if (a[j] && b[i+j] && c[i-j])
      {
         x[i]  = j;  a[j]  = 0;
         b[i+j] = 0; c[i-j] =  0;
         if (i < 8 )
            Try (i+1);
         else
            Xuất(x);
         /* Sau khi in 1 lời giải xong,trả lại
            tình trạng ban đầu còn trống cho hàng
            a[j], đường chéo i+j và đường chéo
            i-j, để tìm lời giải khác */
         a[ j ]  = 1; b[i+j] =  1;  c[i-j] =   1;
      }
}
```

**Khởi tạo mảng với số lượng giá trị tương ứng:

Vì :

$1 \le i,j \le 8 \Rightarrow 2 \le i+j \le 16$ Và $-7 \le i - j \le 7$.

Nên ta có thể khai báo :

```
int  x[8],
     a[8],
     b[15],
     c[15];
```

Với các dữ liệu đã cho, thì lệnh đặt quân hậu sẽ thể hiện bởi :

x[i] = j; // đặt quân hậu thứ i trên cột j.

a[j] = 0; //Khi đặt hậu tại cột j , thì cột j không còn trống nữa

b[i+ j] = 0; //Các đường chéo tương ứng không còn trống.

c[i - j] = 0; //Các đường chéo tương ứng không còn trống.

****Lệnh dời quân hậu là:**

Làm cho hàng i và các đường chéo tương ứng trở thành trống

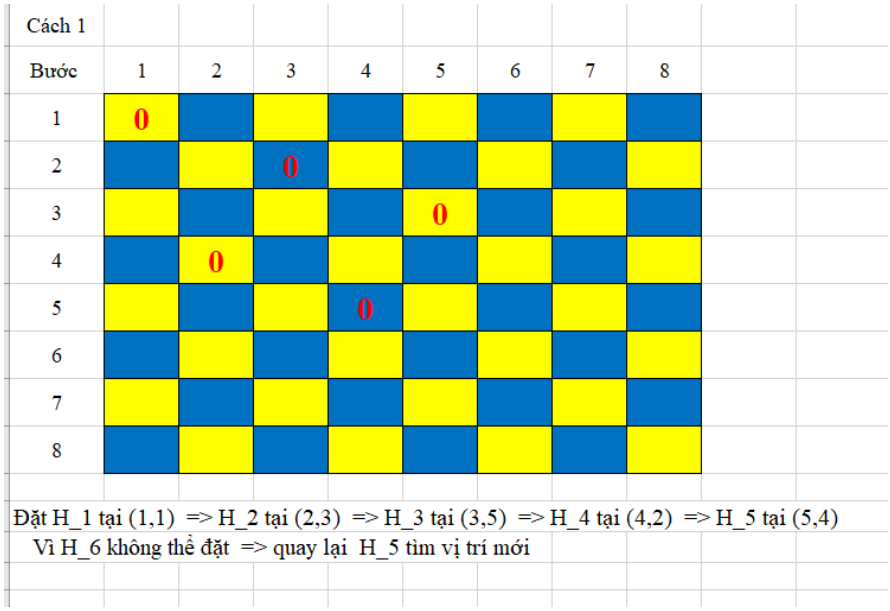
a[j] = 1; b[i+j] = 1; c[i-j] = 1;

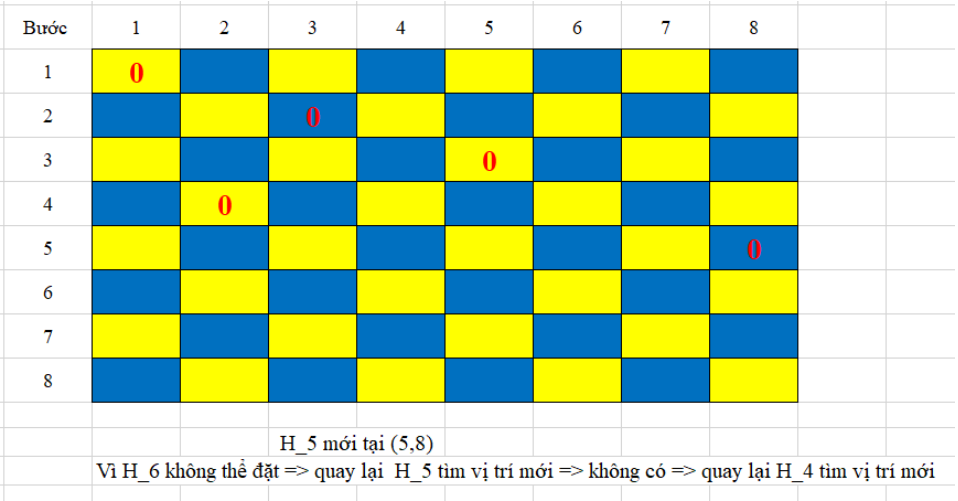
Điều kiện an toàn là ô có tọa độ (i, j) nằm ở hàng và các đường chéo chưa bị chiếm (được thể hiện bằng giá trị True).

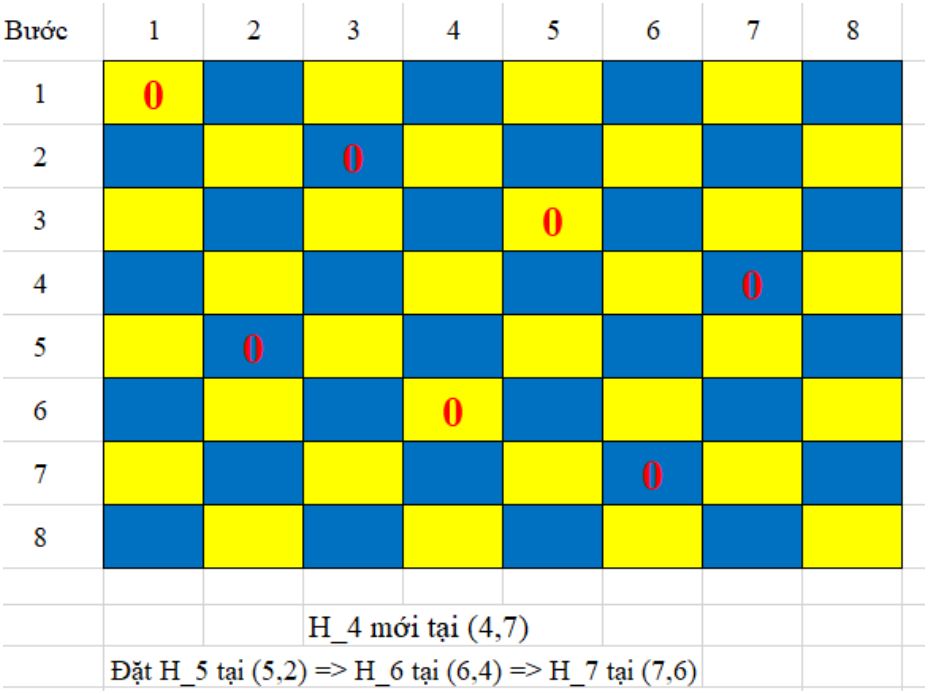
Do đó, có thể được thể hiện bởi biểu thức logic:

(a[j] && b[i+j] && c[i-j])

3. Thực hiện từng bước việc đặt 5, 6, 7 con hậu đầu tiên trên bàn cờ 8x8;







Bước / Cách	1	2	3	4	5	6	7
1	1	5	8	6	3	7	2
2	1	6	8	3	7	4	2
3	1	7	4	6	8	2	5
4	1	7	5	8	2	4	6
5	2	4	6	8	3	1	7
6	2	5	7	1	3	8	6
7	2	5	7	4		1	8
8	2	6	1	7	4	8	3
9	2	6	8	3	1	4	7
10	2	7	3	6	8	5	1
11	2	7	5	8	1	4	6
12	2	8	6	1	3	5	7
13	3	1	7	5	8	2	4

14	**3**	**5**	**2**	**8**	**1**	**7**	**4**
15	**3**	**5**	**2**	**8**	**6**	**4**	**7**
16	**3**	**5**	**7**	**1**	**4**	**2**	**8**
17	**3**	**5**	**8**	**4**	**1**	**7**	**2**
18	**3**	**6**	**2**	**5**	**8**	**1**	**7**
19	**3**	**6**	**2**	**7**	**1**	**4**	**8**
20	**3**	**6**	**2**	**7**	**5**	**1**	**8**
21	**3**	**6**	**4**	**1**	**8**	**5**	**7**
22	**3**	**6**	**4**	**2**	**8**	**5**	**7**
23	**3**	**6**	**8**	**1**	**4**	**7**	**5**
24	**3**	**6**	**8**	**1**	**5**	**7**	**2**
25	**3**	**6**	**8**	**2**	**4**	**1**	**7**
26	**3**	**7**	**2**	**8**	**5**	**1**	**4**
27	**3**	**7**	**2**	**8**	**6**	**4**	**1**
28	**3**	**8**	**4**	**7**	**1**	**6**	**2**
29	**4**	**1**	**5**	**8**	**2**	**7**	**3**
30	**4**	**1**	**5**	**8**	**6**	**3**	**7**
31	**4**	**2**	**6**	**8**	**6**	**1**	**3**
32	**4**	**2**	**7**	**3**	**6**	**8**	**1**
33	**4**	**2**	**7**	**3**	**6**	**8**	**5**
34	**4**	**2**	**7**	**5**	**1**	**8**	**6**
35	**4**	**2**	**8**	**5**	**7**	**1**	**3**
36	**4**	**2**	**8**	**6**	**1**	**3**	**5**
37	**4**	**6**	**1**	**5**	**2**	**8**	**3**
38	**4**	**6**	**8**	**2**	**7**	**1**	**3**
39	**4**	**6**	**8**	**3**	**1**	**7**	**5**
40	**4**	**7**	**1**	**8**	**5**	**2**	**6**
41	**4**	**7**	**3**	**8**	**2**	**5**	**1**
42	**4**	**7**	**5**	**2**	**6**	**1**	**3**

43	**4**	**7**	**5**	**3**	**1**	**6**	**8**
44	**4**	**8**	**1**	**3**	**6**	**2**	**7**
45	**4**	**8**	**1**	**5**	**7**	**2**	**6**
46	**4**	**8**	**5**	**3**	**1**	**7**	**2**
47	**5**	**1**	**4**	**6**	**8**	**2**	**7**
48	**5**	**1**	**8**	**4**	**2**	**7**	**3**
49	**5**	**1**	**8**	**6**	**3**	**7**	**2**
50	**5**	**2**	**4**	**6**	**8**	**3**	**1**
51	**5**	**2**	**4**	**7**	**3**	**8**	**6**
52	**5**	**2**	**6**	**1**	**7**	**4**	**8**
53	**5**	**2**	**8**	**1**	**4**	**7**	**3**
54	**5**	**3**	**1**	**6**	**8**	**2**	**4**
55	**5**	**3**	**1**	**7**	**2**	**8**	**6**
56	**5**	**3**	**8**	**4**	**7**	**1**	**6**
57	**5**	**7**	**1**	**3**	**8**	**6**	**4**
58	**5**	**7**	**1**	**4**	**2**	**8**	**6**
59	**5**	**7**	**2**	**4**	**8**	**1**	**3**
60	**5**	**7**	**2**	**6**	**3**	**1**	**4**
61	**5**	**7**	**2**	**6**	**3**	**1**	**8**
62	**5**	**7**	**4**	**1**	**3**	**8**	**6**
63	**5**	**8**	**4**	**1**	**3**	**6**	**2**
64	**5**	**8**	**4**	**1**	**7**	**2**	**6**
65	**6**	**1**	**5**	**2**	**8**	**3**	**7**
66	**6**	**2**	**7**	**1**	**3**	**5**	**8**
67	**6**	**2**	**7**	**1**	**4**	**8**	**5**
68	**6**	**3**	**1**	**7**	**5**	**8**	**2**
69	**6**	**3**	**1**	**8**	**4**	**2**	**7**
70	**6**	**3**	**1**	**8**	**5**	**2**	**4**
71	**6**	**3**	**5**	**7**	**1**	**4**	**2**

72	**6**	**3**	**5**	**8**	**1**	**4**	**2**
73	**6**	**3**	**7**	**2**	**4**	**8**	**1**
74	**6**	**3**	**7**	**2**	**8**	**5**	**1**
75	**6**	**3**	**7**	**4**	**1**	**8**	**2**
76	**6**	**4**	**1**	**5**	**8**	**2**	**7**
77	**6**	**4**	**2**	**8**	**5**	**7**	**1**
78	**6**	**4**	**7**	**1**	**3**	**5**	**2**
79	**6**	**4**	**7**	**1**	**8**	**2**	**5**
80	**6**	**8**	**2**	**4**	**1**	**7**	**5**
81	**7**	**1**	**3**	**8**	**6**	**4**	**2**
82	**7**	**2**	**4**	**1**	**8**	**5**	**3**
83	**7**	**2**	**6**	**3**	**1**	**4**	**8**
84	**7**	**3**	**1**	**6**	**8**	**5**	**2**
85	**7**	**3**	**8**	**2**	**5**	**1**	**6**
86	**7**	**4**	**2**	**5**	**8**	**1**	**3**
87	**7**	**4**	**2**	**8**	**6**	**1**	**3**
88	**7**	**5**	**3**	**1**	**6**	**8**	**2**
89	**8**	**2**	**4**	**1**	**7**	**5**	**3**
90	**8**	**2**	**5**	**3**	**1**	**7**	**4**
91	**8**	**3**	**1**	**6**	**2**	**5**	**7**
92	**8**	**4**	**1**	**3**	**6**	**2**	**7**

4. Viết chương trình sử dụng C, C++

```
void test(int i) // vong quay lui
{
	int j;
	if (i<n)
	{
		for (j=0; j<n; j++)
		{
			if (a[j] == 1 && b[i+j] ==1 && c[i-j] == 1) //chua co quan hau nao o cot j
```

```
                        {
                                x[i] = j;
                                a[j] = 0; b[i+j] = 0; c[i-j] = 0;
                                test(i+1);  // neu chua xep duoc quan hau thu 8 thi tiep tuc vong lap
                                a[j] = 1; b[i+j] = 1; c[i-j] = 1; // khoi tao lai gia tri cho cot va duong cheo
        }
      }
   }
        else
        {
                print();dem++;
        }
}


void TamHau()
{
        cout << "Nhap so hang (cot): ";
        cin >> n;

        int i;
        for (i=0; i<n; i++)
        {
                a[i] = 1;
                x[i] = -1;
        }
        for (i=0; i<2*n; i++)
        {
                b[i] = 1;
        }
        for(i=1-n; i<n; i++)
        {
                c[i] = 1;
        }
        test(0);
        printf("\n Tong so cach dat quan Hau: %d",dem);
}
```

5. Đánh giá độ phức tạp thuật toán theo lý thuyết: O(n!)