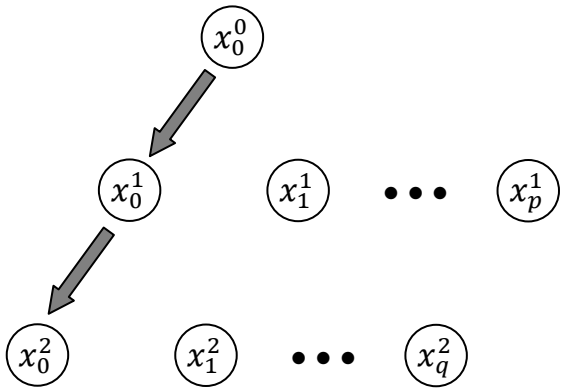
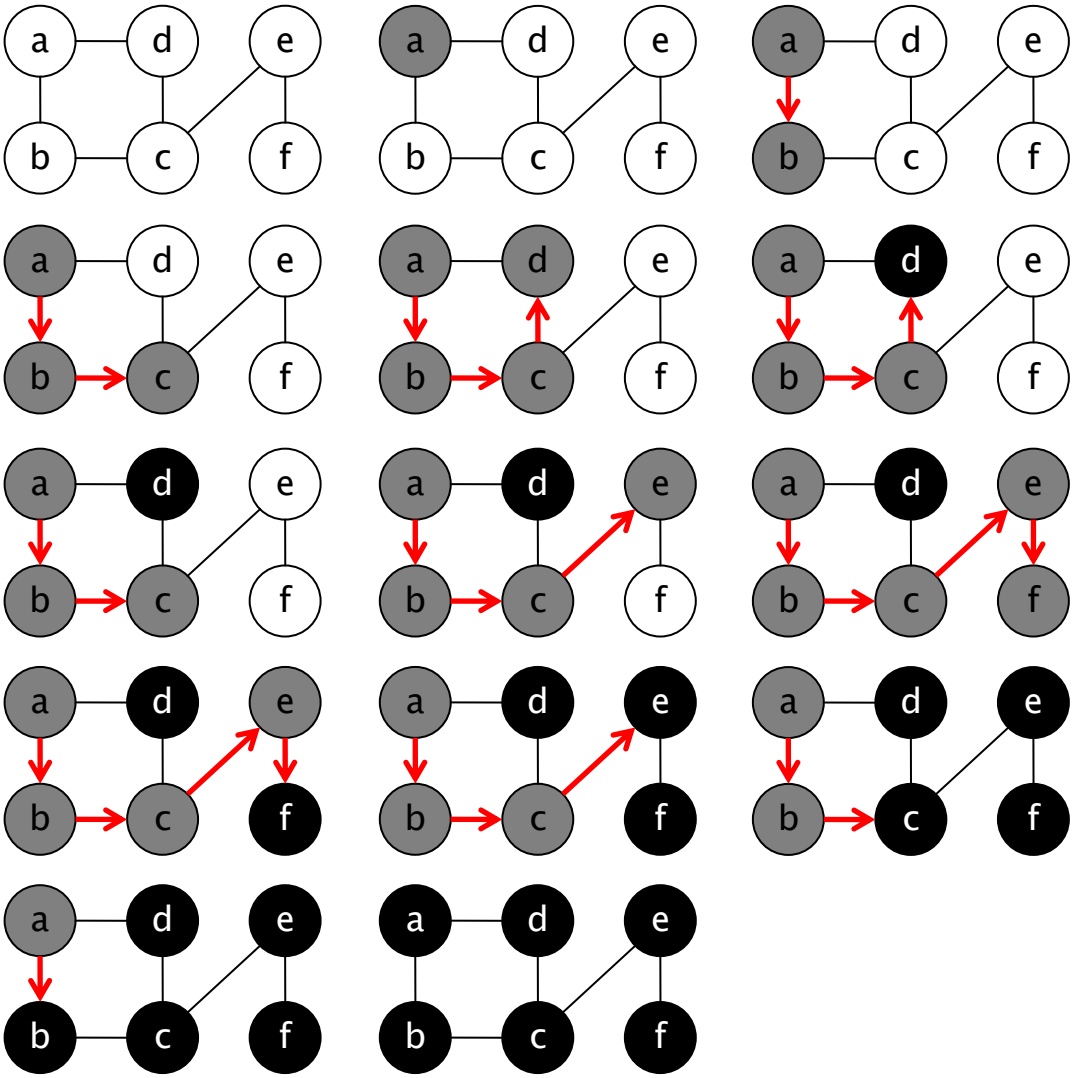
Chương 6: Kỹ thuật Giảm để trị (Decrease-and-Conquer)

Duyệt đồ thị theo chiều sâu (*Depth-First Search* – DFS)

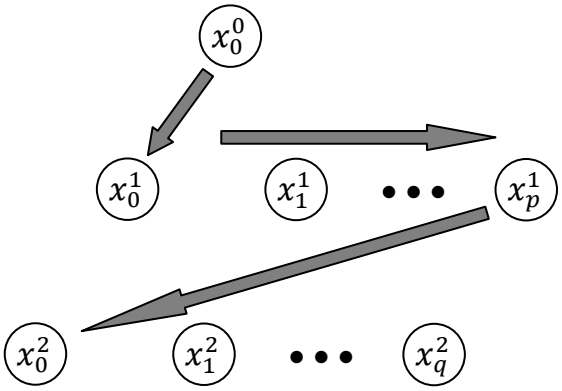


Giải thuật

```
void  DFS(G) {
   for (u ∈ V) {
      color_u = WHITE;
      parent_u = NIL;
   }
   for (u ∈ V)
      if (color_u == WHITE)
         DFS_VISIT(u);
}
void  DFS_VISIT(u ∈ V) {
   color_u = GRAY;
   for (mỗi đỉnh v kề với đỉnh u)
      if (color_v == WHITE) {
         parent_v = u;
         DFS_VISIT(v);
      }
   color_u = BLACK;
}
```

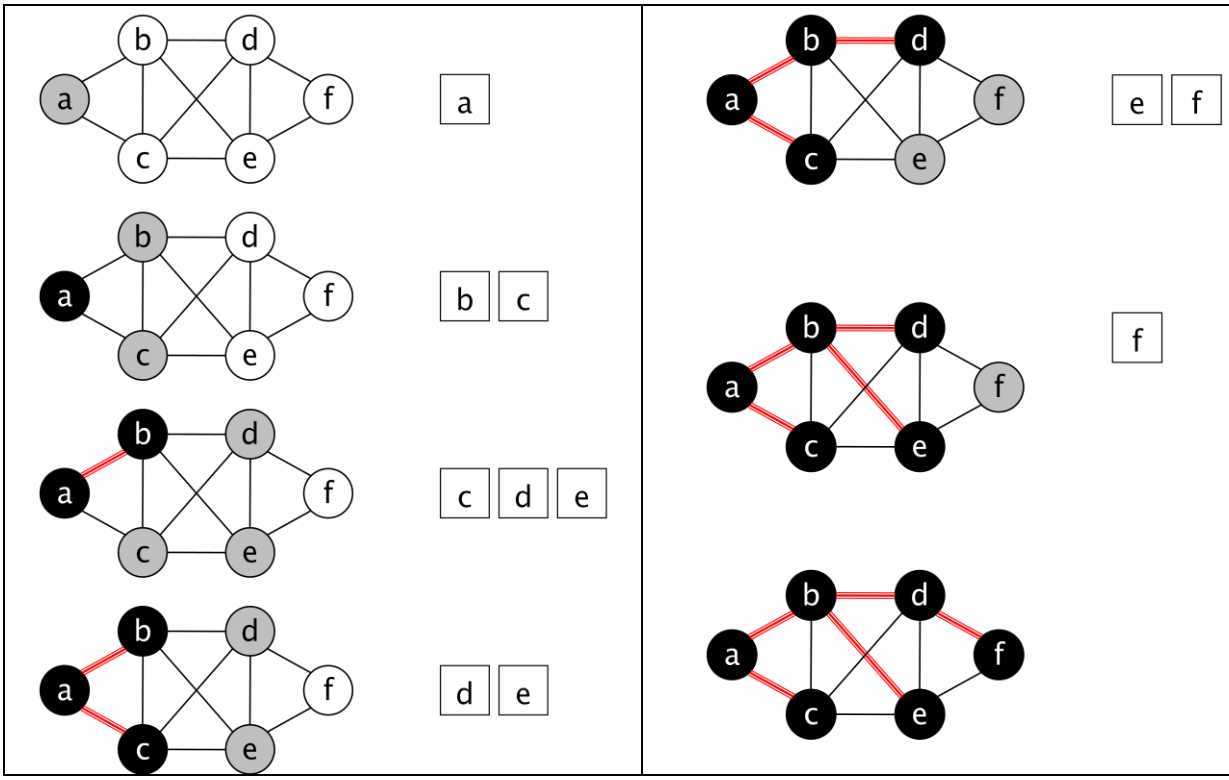


Duyệt đồ thị theo chiều rộng (*Breadth-First Search* – BFS)

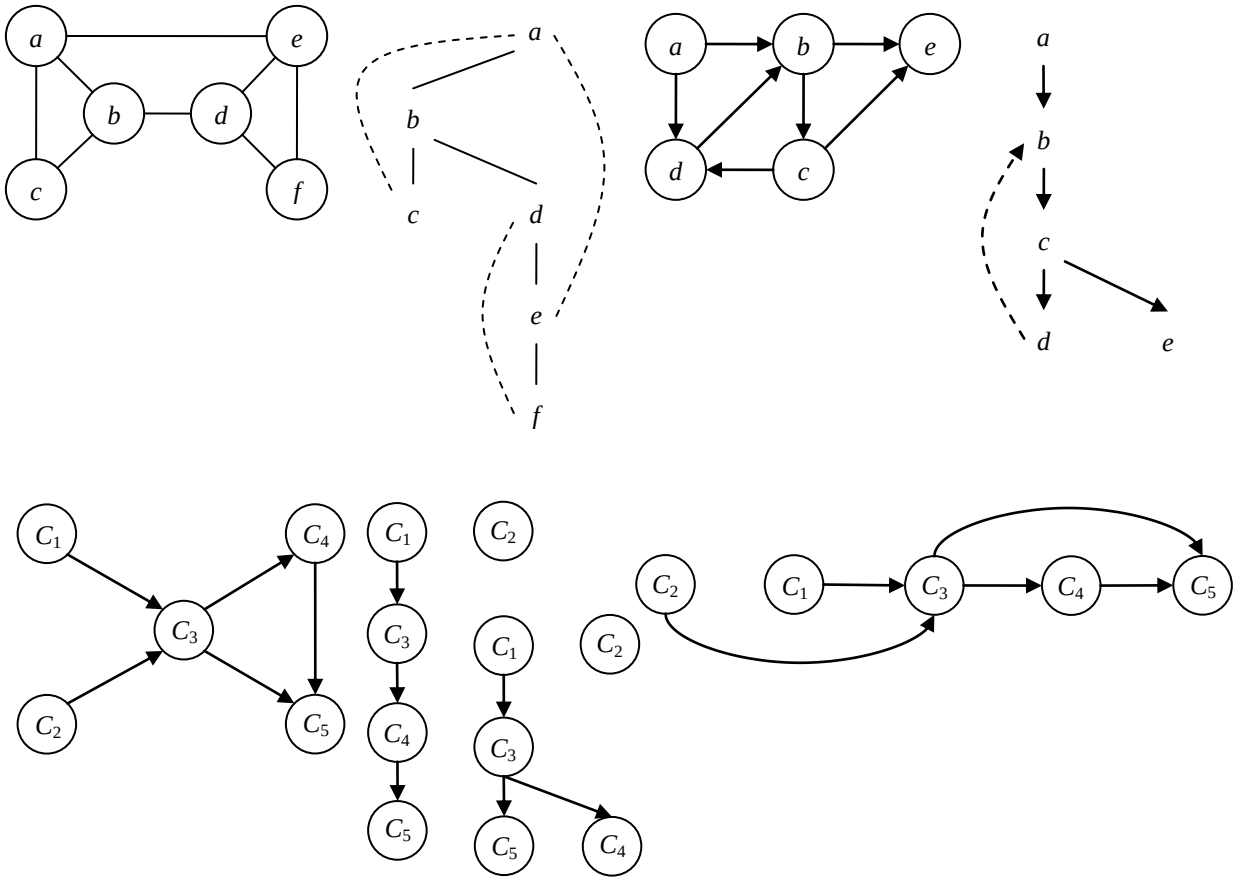


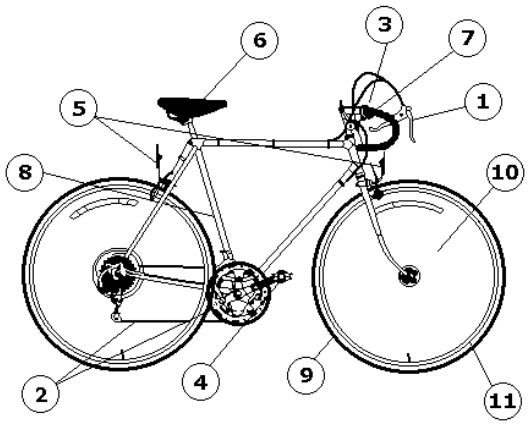
Giải thuật

```
void     BFS(G, root) {
   for (u ∈ V) {
      color_u = WHITE;
      parent_u = NIL;
   }
   color_root = GRAY;
   Q = ∅;
   enQueue(Q, root);
   while (Q ≠ ∅) {
      u = deQueue(Q);
      for (mỗi đỉnh v kề đỉnh u)
         if (color_v == WHITE) {
            color_v = GRAY;
            parent_v = u;
            enQueue(Q, v);
         }
      color_u = BLACK;
   }
}
```

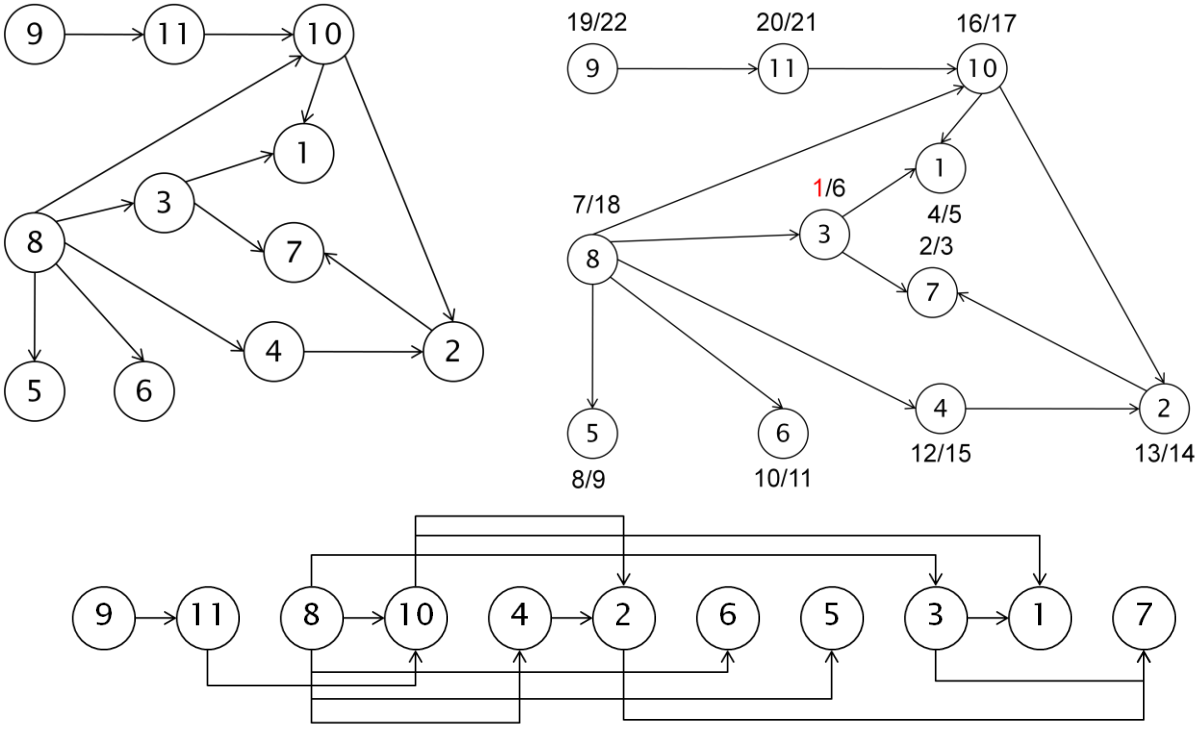


Sắp xếp Topology

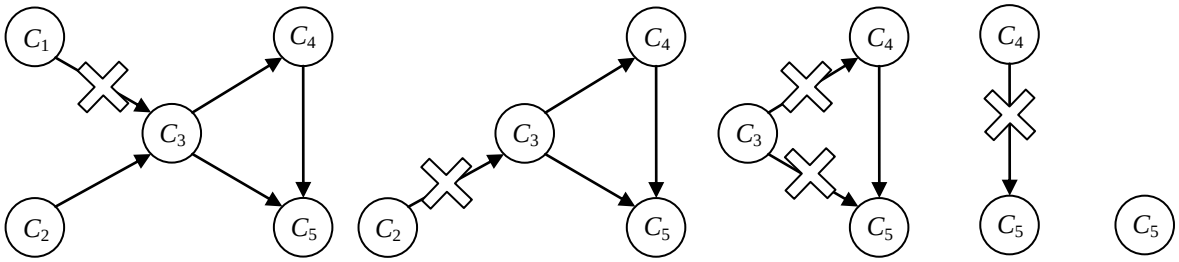




1. Lắp thắng vào tay lái
2. Gắn bộ truyền động
3. Gắn tay lái
4. Lắp bàn đạp và đĩa
5. Lắp đèn xe
6. Lắp yên xe
7. Lắp hộp điều chỉnh tốc độ
8. Lắp (tạo) khung xe
9. Lắp vỏ xe vào vành xe
10. Gắn bánh xe vào khung xe
11. Lắp vành xe vào bánh xe (đúc)



Sắp xếp topo (Giảm để trị và xử lý trực tiếp)



Giải thuật:

```
Topologicalsort(Graph G) {
   Vertex indegree[0 .. |V|] = 0;
   for (mỗi u ∈ V)
      for (mỗi đỉnh v kề với u)
         indegree[v] ++;
   for (mỗi v ∈ V)
      if (indegree[v] == 0)
         enQueue(Q, v);
   while (!isEmpty(Q)) {
      Vertex u = deQueue(Q);
      Output(u);
      for (mỗi đỉnh v kề với u)
         if( -- indegree[v] == 0 )
            enQueue(Q, v);
   }
}
```

Giải thuật Steinhause-Johnson-Trotter

Dãy có 3 phần tử		
1. $\overleftarrow{12\mathbf{3}}$ 2. $\overleftarrow{1\mathbf{3}2}$	3. $\overleftarrow{\mathbf{3}12} \rightarrow \overleftarrow{31\mathbf{2}}$ 4. $\overrightarrow{3\mathbf{2}1} \rightarrow \overrightarrow{321} \rightarrow \overrightarrow{\mathbf{3}21}$	5. $\overrightarrow{2\mathbf{3}1}$ 6. $\overrightarrow{21\mathbf{3}} \rightarrow \overrightarrow{213}$
Dãy có 4 phần tử		
1. $\overleftarrow{123\mathbf{4}}$ 2. $\overleftarrow{12\mathbf{4}3}$ 3. $\overleftarrow{1\mathbf{4}23}$ 4. $\overleftarrow{\mathbf{4}123} \rightarrow \overleftarrow{4123} \rightarrow \overleftarrow{412\mathbf{3}}$ 5. $\overleftarrow{41\mathbf{3}2} \rightarrow \overleftarrow{4132} \rightarrow \overleftarrow{\mathbf{4}132}$ 6. $\overleftarrow{1\mathbf{4}32}$ 7. $\overleftarrow{13\mathbf{4}2}$ 8. $\overleftarrow{132\mathbf{4}} \rightarrow \overleftarrow{1324} \rightarrow \overleftarrow{13\mathbf{2}4}$	9. $\overleftarrow{\mathbf{3}124} \rightarrow \overleftarrow{3124} \rightarrow \overleftarrow{312\mathbf{4}}$ 10. $\overleftarrow{31\mathbf{4}2}$ 11. $\overleftarrow{3\mathbf{4}12}$ 12. $\overleftarrow{\mathbf{4}312} \rightarrow \overleftarrow{4312} \rightarrow \overleftarrow{431\mathbf{2}}$ 13. $\overrightarrow{43\mathbf{2}1} \rightarrow \overrightarrow{4321} \rightarrow \overrightarrow{\mathbf{4}321}$ 14. $\overrightarrow{3\mathbf{4}21}$ 15. $\overrightarrow{32\mathbf{4}1}$ 16. $\overrightarrow{321\mathbf{4}} \rightarrow \overrightarrow{3214} \rightarrow \overrightarrow{\mathbf{3}214}$	17. $\overleftrightarrow{2\mathbf{3}14} \rightarrow \overleftrightarrow{2314} \rightarrow \overleftrightarrow{231\mathbf{4}}$ 18. $\overleftrightarrow{23\mathbf{4}1}$ 19. $\overleftrightarrow{2\mathbf{4}31}$ 20. $\overleftrightarrow{\mathbf{4}231} \rightarrow \overleftrightarrow{4231} \rightarrow \overleftrightarrow{42\mathbf{3}1}$ 21. $\overleftrightarrow{421\mathbf{3}} \rightarrow \overleftrightarrow{4213} \rightarrow \overleftrightarrow{\mathbf{4}213}$ 22. $\overleftrightarrow{2\mathbf{4}13}$ 23. $\overleftrightarrow{21\mathbf{4}3}$ 24. $\overleftrightarrow{213\mathbf{4}} \rightarrow \overleftrightarrow{2134}$

Chương trình

```
void  SJT(int a[], int pos[], int dir[]) {
   int k, i;
   while (true) {
      k = N;
      PrintIt(a);
      while (a[pos[k] + dir[k]] > k) {
         dir[k] = -dir[k];
         if (--k == 0)
            return;
      }
      i = pos[k];
      a[i] = a[i + dir[k]];
      a[i + dir[k]] = k;

      pos[k] = i + dir[k];
      pos[a[i]] = i;
   }
}
void main() {
   int i;
   int a[N + 2], pos[N + 2], dir[N + 2];
   for (i = 1; i <= N; i++) {
      a[i] = pos[i] = i;
      dir[i] = -1;
   }
   a[0] = a[N + 1] = N + 1;
   SJT(a, pos, dir);
}
```

Thứ tự từ điển học (lexicographic order)

Giải thuật (Hàm tìm dãy hoán vị kế tiếp dãy được truyền vào)

```
NextPerm(a[1 .. n]) {
   k = n - 1;
   while (a[k] > a[k+1]) {
      k--;
      if (k == 0)
         return;
   }
   i = n;
   while (a[k] > a[i])
      i--;
   swap(a[k], a[i]);
   r = n;
   s = k + 1;
   while (r > s) {
      swap(a[r], a[s]);
      r--;
      s++;
   }
}
```

Phát sinh tập con

Ví dụ: Phát sinh tập lũy thừa của tập $\{a_1, a_2, a_3\}$

N	Các tập con							
0	$\varnothing$							
1	$\varnothing$	$\{a_1\}$						
2	$\varnothing$	$\{a_1\}$	$\{a_2\}$	$\{a_1, a_2\}$				
3	$\varnothing$	$\{a_1\}$	$\{a_2\}$	$\{a_1, a_2\}$	$\{a_3\}$	$\{a_1, a_3\}$	$\{a_2, a_3\}$	$\{a_1, a_2, a_3\}$

Chương trình (đệ qui)

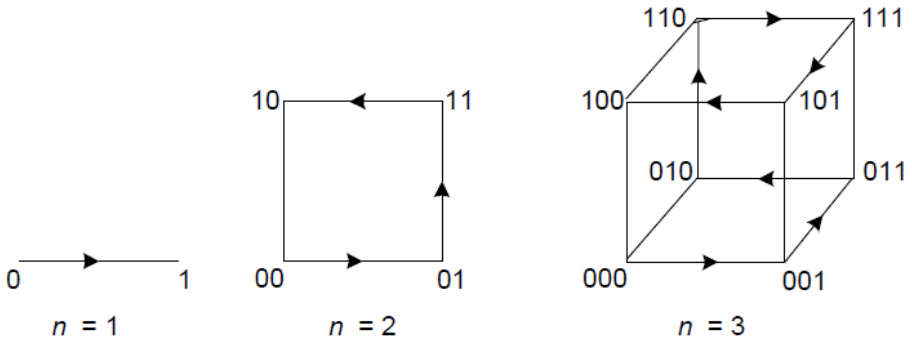
```
void Subsets(int n, int a[]) {
   if (n == N)
      PrintIt(a);
   else {
      a[n] = 0;
      Subsets(n + 1, a);

      a[n] = 1;
      Subsets(n + 1, a);
   }
}
void main () {
   int  a[N];      // Co the khai bao toan cuc
   Subsets(0, a);
}
```

Chương trình (không đệ qui)

```
#include <stdio.h>
const int N = 3;
void main () {
   int  a[N] = {0}, k;
   do {
      PrintIt(a);
      k = N - 1;
      while (k >= 0 && a[k] == 1)
         k--;
      if (k >= 0) {
         a[k] = 1;
         for (k++; k < N; k++)
            a[k] = 0;
      }
   } while (k >= 0);
}
```

Gray code

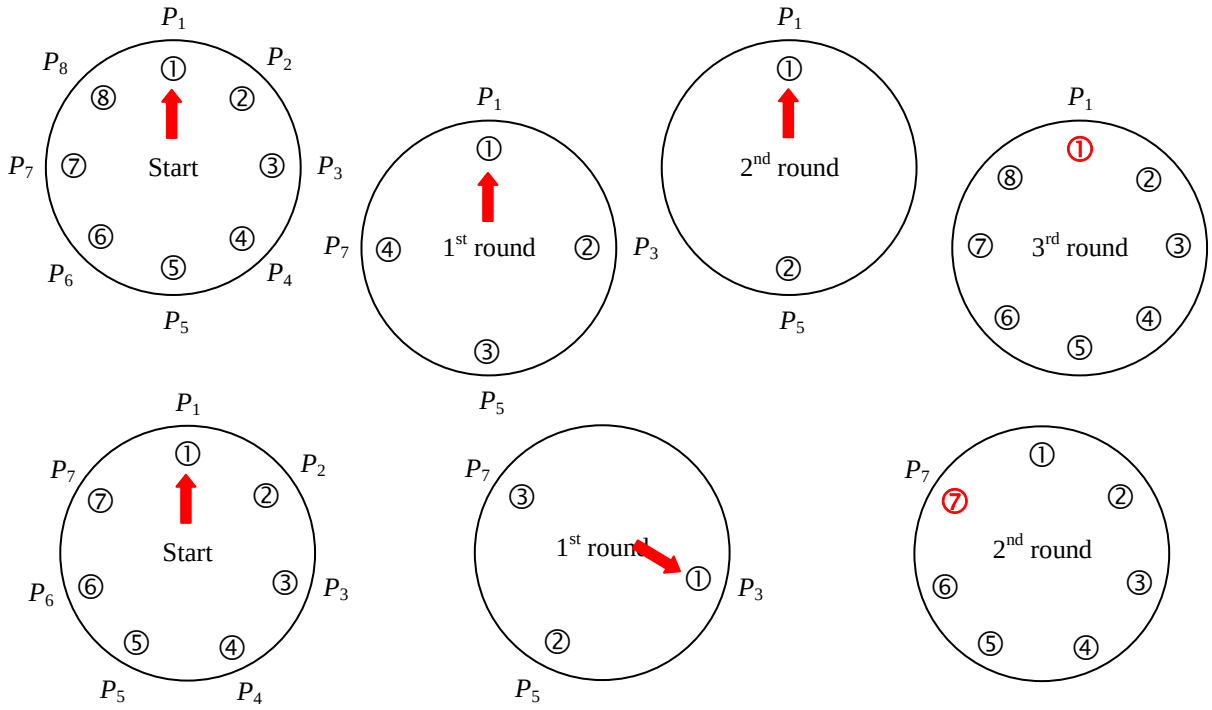


Cách nhân của tá điền Nga

Chương trình (đệ qui)

```
#include <stdio.h>
int   Russ(int n, int m) {
   if (n == 1)
      return m;
   if (n % 2 == 0)
      return Russ(n / 2, m * 2);
   return Russ((n - 1) / 2, m * 2) + m;
}
```

Bài toán Josephus



Vấn đề chọn (*Selection problem*)

Chương trình (đệ qui, không sử dụng mảng phụ)

```
int   Partition(int a[], int left, int right) {
   if (left == right)   return      left;
   int p = a[left];
   int i = left;
   int j = right + 1;
   do {
      do i++; while (a[i] < p);
      do j--; while (a[j] > p);
      swap(a[i], a[j]);
   } while (i < j);
   swap(a[i], a[j]);
   swap(a[left], a[j]);
   return   j;
}
int   Selection(int a[], int left, int right, int k) {
   int pivot = Partition(a, left, right);
   if (left + k - 1 < pivot)
      return   Selection(a, left, pivot - 1, k);
   else
      if (pivot < left + k - 1)
         return   Selection(a, pivot + 1, right, left + k - 1 - pivot);
      else
         return   a[pivot];
}
```

```
void  main() {
   int   a[N + 1], median, k;
   k = (N % 2) ? (N + 1) / 2 : N / 2 + 1;
   a[N] = ∞;
   median = Selection(a, 0, N - 1, k);
}
```

Tìm kiếm nội suy