Chương 3: Kỹ thuật "Brute Force"

Sắp xếp nổi bọt

```
Bubblesort(a[0 .. n – 1]) {
   for (i = 1; i < n; i++)
      for (j = n - 1; j >= i; j--)
         if (a[j - 1] > a[j])
            swap(a[j - 1], a[j]);
}
```

So trùng chuỗi tuần tự

```
SequentialStringSearch(T[0 .. n - 1], P[0 .. m - 1]) {
   for (i = 0; i <= n - m; i++) {
      j = 0;
      while (j < m) && (P[j] == T[i + j])
         j++;
      if (j == m)
         return   i;
   }
   return -1;
}
```

Cặp (điểm) gần nhất

```
BruteForceClosestPoints(P) {
   dmin = ∞;
   for (i = 0; i < n - 1; i++)
      for (j = i + 1; j < n; j++) {
         d = sqrt((xi - xj)^2 + (yi - yj)^2);
         if (d < dmin) {
            dmin = d;
            point1 = i;
            point2 = j;
         }
      }
   return   point1, point2;
}
```

Bao đóng lồi

Giải thuật ($O(n^3)$)

```
for (Tất cả điểm p_i trong tập S: i = 1 → n - 1)
   for (Tất cả điểm q_j trong tập S: j = i + 1 → n) {
      Xây dựng đường thẳng p_i q_j;
      if (Tất cả các điểm khác trong S nằm về một phía của đường p_i q_j)
         Bổ sung đoạn p_i q_j vào danh sách kết quả
   }
```

Giải thuật ($O(n^2)$)

```
findNextExtremePoint(S, cur, curAngle) {
   double minAngle = 2 * Pi;
   S \= cur;
   for (each point p in S) {
      angle = computeAngle(cur, p);
      if (angle < minAngle && angle >= curAngle) {
         Point next = p;
         minAngle = angle;
      }
   }
   S ∪= cur;
   return [next, minAngle];
}
computeConvexHull(S) {
   convexHull = ∅;
   Point first = Điểm có tung độ nhỏ nhất trong S;
   convexHull ∪= first; // Cần đảm bảo thứ tự (thêm vào cuối danh sách)
   curAngle = 0;
   Point cur = first;
   while (true) {
      Point [cur, curAngle] = findNextExtremePoint(S, cur, curAngle);
      convexHull ∪= cur;
      if (first == next)
         break;
   }
   return convexHull;
}
```

Giải thuật tạo tập hợp của các tập con từ tập có kích thước n

```
for (k = 0; k < 2^n; k++)
   In chuỗi bit chiều dài n biểu diễn k;
```

Giải thuật tạo hoán vị

```
taohoanvi(pivot, a, n) {
   if (pivot == n - 1)
      inhoanvi(a, n);
   else
      for (int i = pivot; i < n; i++) {
         hoanvi(a[pivot], a[i]);
         taohoanvi(pivot + 1, a, n);
         hoanvi(a[pivot], a[i]);
      }
}
taohoanvi(0, a, n);
```