Meet in the Middle

Meet in the Middle

You are given an array of n numbers. In how many ways can you choose a subset of the numbers with sum x?

1 ≤ n ≤ 40

Divide into two part then brute force and then use set to find if sum exists or not

String problem

You are given two string S and T of length N. Figure out if we can get string T starting from string S and applyling 4 susbstring reverse algorithms

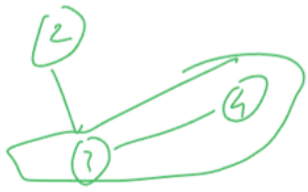
Apply two substring reverse on s and add to set and the apply two on t and add to set

```
for(int l1=0;l1<n;l1++){
    for(int r1=l1;r1<n;r1++){
        for(int l2=0;l2<n;l2++)
            for(int r2=l2;r2<n;r2++)
            string temp=s;
            reverse(temp.begin()+l1,temp.begin()+r1+1);
            reverse(temp.begin()+l2,temp.begin()+r2+1);
            add to set
    }
}
```

Graph problem

Find minimum vertex cover when n<=30 and m<=40

```cpp
// <graph>
bool check(int graph, int vertexCover, int edges[][2], int M) {
  for (int i = 1; i <= M; i++) {
    int a = edges[i][0];
    int b = edges[i][1];
    if (((graph >> a) & 1) == 0 or ((graph >> b) & 1) == 0) {
      // this edge doesn't lie in graph
      continue;
    }
    if (((vertexCover >> a) & 1) == 0 && ((vertexCover >> b) & 1) == 0) {
      // no edge vertex lies in vertex cover
      return false;
    }
  }
  return true;
}
```



```cpp
int solve(int edges[][2], int N, int M) {
  unacademy 1e9;
  int leftSize = N / 2;
  int rightSize = N - leftSize;

  vector<int> adj[N + 5];
  for (int i = 1; i <= M; i++) {
    adj[edges[i][0]].push_back(edges[i][1]);
    adj[edges[i][1]].push_back(edges[i][0]);
  }

  int dp[(1 << leftSize) + 5];

  for (int graph = 0; graph < (1 << leftSize); graph++) {
    int minVertexConver = inf;

    if (check(graph, 0, edges, M))
      minVertexConver = 0;
    for (int vertexCover = graph; vertexCover > 0;
         vertexCover = (vertexCover - 1) & graph) {
      if (check(graph, vertexCover, edges, M)) {
        minVertexConver = min(minVertexConver, countSetBits(vertexCover));
      }
    }
    dp[graph] = minVertexConver;
  }
  int res = inf;
  for (int vertexCover = 0; vertexCover < (1 << rightSize); vertexCover++) {
    int graph = (1 << rightSize) - 1;
    if (!check(graph << leftSize, vertexCover, edges, M))
      continue;

    int rightVertexCover = countSetBits(vertexCover);
    int remGraph = graph ^ vertexCover;

    int leftVerticesToTake = 0;

    for (int remVertex = 0; remVertex < rightSize; remVertex++) {
      if ((remGraph >> remVertex) & 1) {
        int actualVertex = remVertex + leftSize;
        for (int &leftVertex : adj[actualVertex]) {
          if (leftVertex < leftSize)
            leftVerticesToTake |= (1 << leftVertex);
        }
      }
    }

    int leftVertexCover = countSetBits(leftVerticesToTake) +
                          dp[((1 << leftSize) - 1) ^ leftVerticesToTake];
    res = min(res, leftVertexCover + rightVertexCover);
  }
  return res;
}
```