Here we eliminate the randomization part of a randomized algorithm and turn it into a deterministic algorithm.

- We take the Max-Cut problem and derandomize the previous approximation algorithm.

- The derandomization is achieved by conditional expectations.

- Let $B_i$ be the set where a vertex $v_i$ is placed in the Max-Cut approximation algorithm.

- Suppose we place $v_i$ deterministically (either to A or B) and have already placed first $k$ vertices $v_1, v_2, \ldots, v_k$. 
\( E[C(A,B) \mid x_1, x_2, \ldots, x_k] \): Expected value of the cut after we have placed \( k \) vertices \( x_1, x_2, \ldots, x_k \) and the remaining vertices are placed randomly.

Our goal is to show how to place the next vertex so that following holds inductively:

1. \( E[C(A,B) \mid x_1, x_2, \ldots, x_k] \leq E[C(A,B) \mid x_1, \ldots, x_{k+1}] \)

with the base case

2. \( E[C(A,B)] = E[C(A,B) \mid x_1] \).

Then, we have

3. \( E[C(A,B)] \leq E[C(A,B) \mid x_1, x_2, \ldots, x_k] \).

Observation: The RHS of 3 is the value of the cut determined by the algorithm which places vertices deterministically. Since \( E[C(A,B)] \geq m/2 \), we have a cut of size at least \( m/2 \).
Now, we show how one can ensure

- The base case $E[C(A,B) | x_1] = E[C(A,B)]$ is trivially true because it does not matter where we place the first vertex.

- Inductive step:
  - Consider placing $x_{k+1}$ randomly so that $x_{k+1} = A$ or $B$ with prob. $\frac{1}{2}$

$$E[C(A,B) | x_1, x_2, \ldots, x_k] = \frac{1}{2} E[C(A,B) | x_1, x_2, \ldots, x_{k+1} = A] + \frac{1}{2} E[C(A,B) | x_1, x_2, \ldots, x_{k+1} = B]$$

  - $\max(E[C(A,B) | x_1, x_2, \ldots, x_{k+1} = A], E[C(A,B) | x_1, x_2, \ldots, x_{k+1} = B]) \geq E[C(A,B) | x_1, x_2, \ldots, x_k]$.

- So we place $x_{k+1}$ into the set that provides larger expectation giving $E[C(A,B) | x_1, x_2, \ldots, x_k] \leq E[C(A,B) | x_1, x_2, \ldots, x_{k+1}]$. 

\[\frac{1}{2} E[C(A,B) | x_1, x_2, \ldots, x_{k+1} = A] + \frac{1}{2} E[C(A,B) | x_1, x_2, \ldots, x_{k+1} = B] \]
Computing $E[C(A,B) | x_1, x_2, \ldots x_{k+1}=A]$:

- Conditioning gives the placement of first $k+1$ vertices
- We can compute the number of edges among these vertices that are in the cut
- For all remaining edges, the prob. that it will later contribute to the cut is $\frac{1}{2}$ since its two endpoints are placed in different sets with prob. $\frac{1}{2}$.

- By linearity of expectation $E[C(A,B) | x_1, x_2, \ldots x_{k+1}=A]$ is the #edges in the cut whose endpoints are in the first $k+1$ vertices plus half of the remaining edges.

- The above expectation can be computed in linear time for both $x_{k+1}=A$ and $x_{k+1}=B$. 
So, the algorithm chooses $v_{k+1}$ to be in A or B depending on which expectation is larger.

In fact, the two expectations differ only by the quantity: # of neighbors $v_{k+1}$ has in A or B.

**Algorithm:**

- Order the vertices arbitrarily.
- Place the first vertex into any of the two sets.
- Place each successive vertex into the set that maximizes the cut size, i.e., place each vertex into the set that has fewer neighbors, breaking the ties arbitrarily.