Let $\mathcal{L}$ be an instance of LP with $n$ variables and $m$ constraints. Then we have the following:

A. $\mathcal{L}$ is always feasible.

B. $\mathcal{L}$ might not be feasible, but it can be made feasible by changing the value of one of the variables.

C. $\mathcal{L}$ might not be feasible, but can be fixed by adding a single variable with the appropriate value.

D. $\mathcal{L}$ might not be feasible, but can be fixed by adding two variable with the correct value (one need two variables because of the equality constraints).

E. $\mathcal{L}$ might not be feasible, and this can not be fixed.
19.1: The Simplex Algorithm in Detail
Simplex algorithm

Simplex( $\hat{L}$ a LP )
Transform $\hat{L}$ into slack form.
Let $L$ be the resulting slack form.
$L' \leftarrow \text{Feasible}(L)$
$x \leftarrow \text{LPStartSolution}(L')$
$x' \leftarrow \text{SimplexInner}(L', x)$ (*)
$z \leftarrow$ objective function value of $x'$
if $z > 0$ then
    return “No solution”
$x'' \leftarrow \text{SimplexInner}(L, x')$
return $x''$
Simplex algorithm...

1. **SimplexInner**: solves a LP if the trivial solution of assigning zero to all the nonbasic variables is feasible.

2. \( L' = \text{Feasible}(L) \) returns a new LP with feasible solution.

3. Done by adding new variable \( x_0 \) to each equality.

4. Set target function in \( L' \) to \( \text{min } x_0 \).

5. original LP \( L \) feasible \( \iff \) LP \( L' \) has feasible solution with \( x_0 = 0 \).

6. Apply **SimplexInner** to \( L' \) and solution computed (for \( L' \)) by \( \text{LPStartSolution}(L') \).

7. If \( x_0 = 0 \) then have a feasible solution to \( L \).

8. Use solution in **SimplexInner** on \( L \).

9. need to describe **SimplexInner**: solve LP in slack form given a feasible solution (all nonbasic vars assigned value 0).
Notations

$B$ - Set of indices of basic variables
$N$ - Set of indices of nonbasic variables
$n = |N|$ - number of original variables
$b, c$ - two vectors of constants
$m = |B|$ - number of basic variables (i.e., number of inequalities)
$A = \{a_{ij}\}$ - The matrix of coefficients
$N \cup B = \{1, \ldots, n + m\}$
$v$ - objective function constant.

LP in slack form is specified by a tuple $(N, B, A, b, c, v)$. 
The corresponding LP

\[
\text{max } z = v + \sum_{j \in N} c_j x_j,
\]

s.t. \( x_i = b_i - \sum_{j \in N} a_{ij} x_j \) for \( i \in B \),

\( x_i \geq 0, \quad \forall i = 1, \ldots, n + m. \)
max \quad z = 29 - \frac{1}{9}x_3 - \frac{1}{9}x_5 - \frac{2}{9}x_6

\begin{align*}
x_1 &= 8 + \frac{1}{6}x_3 + \frac{1}{6}x_5 - \frac{1}{3}x_6 \\
x_2 &= 4 - \frac{8}{3}x_3 - \frac{2}{3}x_5 + \frac{1}{3}x_6 \\
x_4 &= 18 - \frac{1}{2}x_3 + \frac{1}{2}x_5
\end{align*}
19.2: The SimplexInner Algorithm
Description \textbf{SimplexInner} algorithm:

1. \textbf{LP} is in slack form.

2. Trivial solution $x = \tau$ (i.e., all nonbasic variables zero), is feasible.

3. Objective value for this solution is $v$.

4. Reminder: Objective function is $z = v + \sum_{j \in N} c_j x_j$.

5. $x_e$: nonbasic variable with positive coefficient in objective function.

6. Formally: $e$ is one of the indices of $\{ j \mid c_j > 0, j \in N \}$.

7. $x_e$ is the \textbf{entering variable} (enters set of basic variables).

8. If increase value $x_e$ (from current value of 0 in $\tau$)...

9. ... one of basic variables is going to vanish (i.e., become zero).
Choosing the leaving variable

1. \( x_e \): entering variable
2. \( x_l \): leaving variable – vanishing basic variable.
3. Increase value of \( x_e \) till \( x_l \) becomes zero.
4. How do we now which variable is \( x_l \)?
5. Set all nonbasic to 0 zero, except \( x_e \)
6. \( x_i = b_i - a_{ie} x_e \), for all \( i \in B \).
7. Require: \( \forall i \in B \quad x_i = b_i - a_{ie} x_e \geq 0 \).
8. \( \Rightarrow x_e \leq \left( b_i / a_{ie} \right) \)
9. \( l = \arg \min_i b_i / a_{ie} \)
10. If more than one achieves \( \min_i b_i / a_{ie} \), just pick one.
Pivoting on $x_e$...

1. Determined $x_e$ and $x_l$.

2. Rewrite equation for $x_l$ in LP.
   
   1. (Every basic variable has an equation in the LP!)
   2. $x_l = b_l - \sum_{j \in N} a_{lj} x_j$

   $$\implies x_e = \frac{b_l}{a_{le}} - \sum_{j \in N \cup \{l\}} \frac{a_{lj}}{a_{le}} x_j,$$
   where $a_{ll} = 1$.

3. Cleanup: remove all appearances (on right) in LP of $x_e$.

4. Substituting $x_e$ into the other equalities, using above.

5. Alternatively, do Gaussian elimination remove any appearance of $x_e$ on right side LP (including objective). Transfer $x_l$ on the left side, to the right side.
Pivoting continued...

1. End of this process: have new *equivalent* LP.
2. basic variables: \( B' = (B \setminus \{l\}) \cup \{e\} \)
3. non-basic variables: \( N' = (N \setminus \{e\}) \cup \{l\} \).
4. End of this *pivoting* stage: LP objective function value increased.
5. Made progress.
6. LP is completely defined by which variables are basic, and which are non-basic.
7. Pivoting never returns to a combination (of basic/non-basic variable) already visited.
8. ...because improve objective in each pivoting step.
9. Can do at most \( \binom{n+m}{n} \leq \left( \frac{n+m}{n} \cdot e \right)^n \).
10. examples where \( 2^n \) pivoting steps are needed.
Simplex algorithm summary...

1. Each pivoting step takes polynomial time in $n$ and $m$.
2. Running time of Simplex is exponential in the worst case.
3. In practice, Simplex is extremely fast.
Pivoting with zeroes?

Consider a pivoting step, with $x_e$ as the entering variable, and $x_\ell$ as the leaving variable, with the relevant constraint in the LP being:

$$x_\ell = 0 - \sum_{j \in N} a_{lj} x_j.$$

(A) Doing the pivoting step would involve division by zero, and as such the Simplex algorithm would fail.

(B) There is no problem.

(C) In an LP the constant in a constraint can never be zero, so this is an impossible scenario.

(D) If there is any problem, it can be solved by choosing a different entering/leaving variables.

(E) The pivoting step would not improve the LP objective function. Simplex might pivot in a loop forever.
Degeneracies

1. **Simplex** might get stuck if one of the $b_i$s is zero.
2. More than $> m$ hyperplanes (i.e., equalities) passes through the same point.
3. Result: might not be able to make any progress at all in a pivoting step.
4. Solution I: add tiny random noise to each coefficient. Can be done symbolically. Intuitively, the degeneracy, being a local phenomena on the polytope disappears with high probability.
Degeneracies – cycling

1 Might get into cycling: a sequence of pivoting operations that do not improve the objective function, and the bases you get are cyclic (i.e., infinite loop).

2 Solution II: **Bland’s rule.**
Always choose the lowest index variable for entering and leaving out of the possible candidates.
(Not prove why this work - but it does.)
19.2.1: Correctness of linear programming
Correctness of LP

**Definition**
A solution to an LP is a **basic solution** if it the result of setting all the nonbasic variables to zero.

**Simplex** algorithm deals only with basic solutions.

**Theorem**

For an arbitrary linear program, the following statements are true:

1. *If there is no optimal solution, the problem is either infeasible or unbounded.*
2. *If a feasible solution exists, then a basic feasible solution exists.*
3. *If an optimal solution exists, then a basic optimal solution exists.*

Proof: is constructive by running the simplex algorithm.
19.2.2: On the ellipsoid method and interior point methods
On the ellipsoid method and interior point methods

1. **Simplex** has exponential running time in the worst case.

2. **Ellipsoid method** is *weakly* polynomial. It is polynomial in the number of bits of the input.


4. In 1984, Karmakar came up with a different method, called the *interior-point method*.

5. Also weakly polynomial. Quite useful in practice.

6. Result in arm race between the interior-point method and the simplex method.

7. **BIG OPEN QUESTION:** Is there *strongly* polynomial time algorithm for linear programming?
Solving LPs without ever getting into a loop - symbolic perturbations

Details in the class notes.