2. Assume there are two language $L_1 \in$ P and $L_2 \in$ P

   Therefore there's a TM M_1 that solves L_1 and TM M_2 that solves L_2 These both run in polynomial time as both are $\in$ P.

   TM M_x = "on input w:

   run w on M_1 if it accepts, accept

   run w on M_2 if it accepts, accept

   else: reject.

   Since we know that M_1 and M_2 run in polynomial time they run in O(n^c) where c is some constant and n = |w| we know that M_x runs in O(n^c) + O(n^c) which is simply O(n^c). therefore since $M_x \in$ P and P is closed under union.

   P is closed under concatenation

   Assume there are two language $L_1 \in$ P and $L_2 \in$ P

   Therefore there's a TM M_1 that solves L_1 and TM M_2 that solves L_2 These both run in polynomial time as both are $\in$ P.

   TM M_x = "on input w:

   1. W can be cut w into two strings, in n different ways
   2. for each cut:

      i. check if $\text{cut}_1 \in L_1$

      ii. check if $\text{cut}_2 \in L_2$

      if both accept, accept.

   Since we know that M_1 and M_2 run in polynomial time they run in O(n^c) where c is some constant and n = |w| we know that M_x rune in O(n^c) + O(n^c) which is simply O(n^c). therefore since $M_x \in$ P and P is closed under concatenation.

   P is closed under complement

   Assume there's a language L_1 with a TM M_1 that decides it $\in$ P

   Then there's a TM M_2 that accepts co-$L_1 \in$ P.

   M_2 = "on input w:

   1. run M_1 on w.

      If M_1 accepts, reject.

      if M_1 rejects, accept.

   Since M_1 is a decider we know that it will always halt in polynomial time and we therefore can just output the opposite in polynomial time. Therefore P is closed under complement