Group anagrams - hash toble
las Solution;
def group_avagrams (self, stis; list(stil) > list(list)
result = default diet (list) for s in strs:
for s in strs;
count = Lo 1 # 26
for char in s;
for char in S; count [ord (char) - ord(")"] +=1 result [tuple (count)]. append (S) return list (zerult values!)
result [tuple (count)]. append (S)
return list (bosult values ())
int (Slution). group-amagrams (Szs=Eact, pots, typ, at, Sop, hot]
Stop 1:
self -> solution instance
Stop 1: self -> solution instance strs -> [oct ') pots', tops', cot', Stop', not"]

Step 2: Lesult -> Eg - default diet < class "list">

Steps: for char in s: char -> "a"

Step 7; for char in s:

 Step 9; for char in si

Step 10: eount [80d ("hob") - 80d ("a")] +=1

Count [80d ("L") - 80d ("a")] +=1

Count [116 - 97] +=1

Count [19] +=1

Count [19] +=1

Count -> [1,0,1,0,0,0,0,0,0,0,0,0] -1,0,0 -1

Step 11: result [tuple (count)].append(c)
result->2(1,0,1,0,0,0,0--1,0,00): ["act"] 3

Step 12: for sin stres:

Step 12: count = [0] # 26

count -> [0,0,0,0 - - 0]

Step 14: for show in s:

Step 15: count [ad("p") - ord ("a")] +=1

Step 16: for char in s
char -> "o"

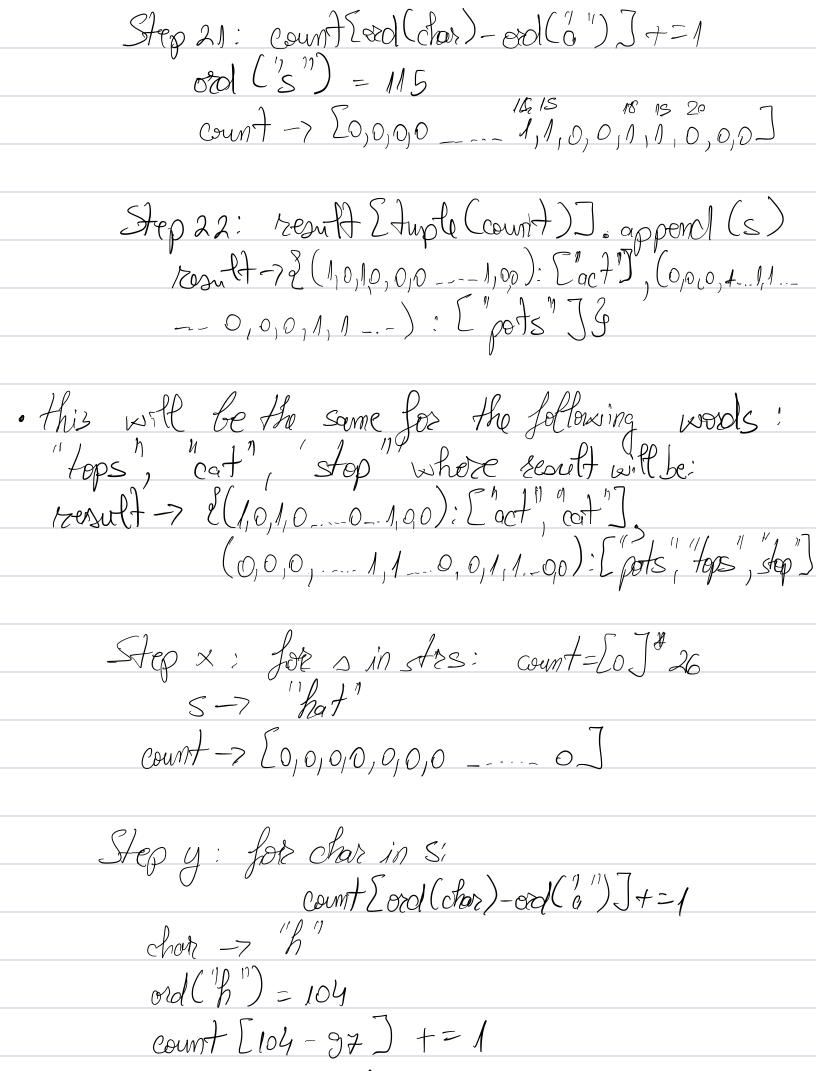
Stop 17: Count [erd(o'') - erd(o'')] + = 1ord('o'') = 111

Count [14] J + = 1 count - > [0,0,0,0 - - - 1, 1,0,0,0]

Step 19: for char in S char -> "t"

Step 19: count 2 ord (2 har) - 80 (2 har) - 80

Step 20: for clor ins char -> 's'



count [0,0,0...1...0,0...]
- for "t" and "o" characters: count [1,0,0, -- 1 -- 0,0,1,0,0 --] result will become -> result Etuple (count)] append (s) (0,0,0,....1,1,0,0,1,1,0,0): [pots, tops, step],
(1,0,0,1,0,0,1,0): [hot]] G Stop 2: list (result. values ()) last sepult -> [[act " act"], [pots", tops " stop], [hat "]]