



GitHub

https://goo.gl/Q7WYVWZ



Universidad de La Laguna

III Congreso de Estudiantes de Ingeniería Informática

CONCATENACIÓN DE LENGUAJES

MIGUEL JIMÉNEZ GOMIS Y PABLO MARTÍN GONZÁLEZ

TUTOR: COROMOTO ANTONIA LEÓN HERNÁNDEZ

Departamento de Ingeniería Informática y Sistemas de la Universidad de La Laguna

DESCRIPCIÓN DEL PROBLEMA

La información del problema proporcionada por la IACM-ICPC describe lo siguiente:
Un lenguaje es un conjunto de cadenas y la concatenación de dos lenguajes es el conjunto resultante de concatenar las cadenas del segundo lenguaje al final de las cadenas del primer lenguaje.
Por ejemplo, si tenemos dos lenguajes A y B tales que:
A = {cat, dog, mouse}
B = {rat, bat}
La concatenación de A y B será:
C = {catrat, catbat, dograt, dogbat, mouserat, mousebat}
Dados dos lenguajes, la tarea consiste solamente en contar el número de cadenas en la concatenación de dichos lenguajes.

Entrada
Pueden haber muchos casos de pruebas. La primera línea de la entrada contendrá el número de casos de prueba, T ($1 \leq T \leq 25$). Después vendrá la descripción de los T casos. La primera línea de cada caso contendrá dos enteros, M y N ($M, N < 1500$), que son el número de cadenas de cada lenguaje. Las siguientes M líneas contienen las cadenas del primer lenguaje y las otras N líneas siguientes te dan el segundo lenguaje. Puedes asumir que esas cadenas están formadas por letras minúsculas ('a' hasta la 'z'), y de una longitud menor a 10 caracteres que estarán presentadas en una línea separada sin ningún espacio alrededor. Las cadenas de la entrada pueden no estar ordenadas y no estarán duplicadas.
Ejemplo entrada:
2
3 2
Cat
Dog
Mouse
Rat
Bat
1 1
abc
cab

Salida
Para cada uno de los casos de prueba se deberá mostrar solamente una línea de salida. La salida de cada test comenzará con el número serial de dicho test, seguido por el número de cadenas en la concatenación del segundo lenguaje tras el primero.
Ejemplo de salida:
Case 1: 6
Case 2: 1

CÓDIGO Y RESULTADOS

```
1 def main():
2     T = int(input())
3     Case = []
4     while (T > 0):
5         result = set()
6         A = []
7         B = []
8         M, N = input().split()
9         M, N = [int(M), int(N)]
10        while (M > 0):
11            A.append(input())
12            M -= 1
13        while (N > 0):
14            B.append(input())
15            N -= 1
16        for x in A:
17            for y in B:
18                dummy = x + y
19                result.add(dummy)
20        Case.append(len(result))
21        T -= 1
22    for i in Case:
23        print("Case " + str(Case.index(i) + 1) + ": " + str(i))
24
25 if __name__ == '__main__':
26     main()
```

Total Submissions		Users that tried it		Users that solved it
10096		1483		988
Your best accepted try				
Ranking	Submission	Run Time	Language	Submission Date
86	20343513	0.400	PYTH3	2017-11-14 09:58:29

CONCLUSIONES

Para la realización de este proyecto decidimos programar usando python3 debido a su facilidad a la hora de escribir el código y la sencillez del mismo. Tras comparar nuestros resultados con los del top del IACM-ICPC, podemos concluir que python, no es el lenguaje más adecuado cuando se quiere optimizar al máximo el tiempo de ejecución de un programa para participar en competiciones de este estilo en las que lo importante es la velocidad, y se compara solo eso no haciendo diferencia por categorías de los diferentes lenguajes. Aunque sencillo, nuestro código es interpretado, y este proceso no puede competir con lenguajes compilados mucho más rápidos a la hora de ejecutar como C o C++, como podemos ver en la siguiente imagen:

Ranking		Submission		Run Time		Language		Submission Date	
1	10096	20343513	0.400	PYTH3	2017-11-14 09:58:29	2	10097	20343514	0.400
3	10098	20343515	0.400	PYTH3	2017-11-14 09:58:30	4	10099	20343516	0.400
5	10100	20343517	0.400	PYTH3	2017-11-14 09:58:31	6	10101	20343518	0.400
7	10102	20343519	0.400	PYTH3	2017-11-14 09:58:32	8	10103	20343520	0.400
9	10104	20343521	0.400	PYTH3	2017-11-14 09:58:33	10	10105	20343522	0.400
11	10106	20343523	0.400	PYTH3	2017-11-14 09:58:34	12	10107	20343524	0.400
13	10108	20343525	0.400	PYTH3	2017-11-14 09:58:35	14	10109	20343526	0.400
15	10110	20343527	0.400	PYTH3	2017-11-14 09:58:36	16	10111	20343528	0.400
17	10112	20343529	0.400	PYTH3	2017-11-14 09:58:37	18	10113	20343530	0.400
19	10114	20343531	0.400	PYTH3	2017-11-14 09:58:38	20	10115	20343532	0.400
21	10116	20343533	0.400	PYTH3	2017-11-14 09:58:39	22	10117	20343534	0.400
23	10118	20343535	0.400	PYTH3	2017-11-14 09:58:40	24	10119	20343536	0.400
25	10120	20343537	0.400	PYTH3	2017-11-14 09:58:41	26	10121	20343538	0.400
27	10122	20343539	0.400	PYTH3	2017-11-14 09:58:42	28	10123	20343540	0.400
29	10124	20343541	0.400	PYTH3	2017-11-14 09:58:43	30	10125	20343542	0.400
31	10126	20343543	0.400	PYTH3	2017-11-14 09:58:44	32	10127	20343544	0.400
33	10128	20343545	0.400	PYTH3	2017-11-14 09:58:45	34	10129	20343546	0.400
35	10130	20343547	0.400	PYTH3	2017-11-14 09:58:46	36	10131	20343548	0.400
37	10132	20343549	0.400	PYTH3	2017-11-14 09:58:47	38	10133	20343550	0.400
39	10134	20343551	0.400	PYTH3	2017-11-14 09:58:48	40	10135	20343552	0.400
41	10136	20343553	0.400	PYTH3	2017-11-14 09:58:49	42	10137	20343554	0.400
43	10138	20343555	0.400	PYTH3	2017-11-14 09:58:50	44	10139	20343556	0.400
45	10140	20343557	0.400	PYTH3	2017-11-14 09:58:51	46	10141	20343558	0.400
47	10142	20343559	0.400	PYTH3	2017-11-14 09:58:52	48	10143	20343560	0.400
49	10144	20343561	0.400	PYTH3	2017-11-14 09:58:53	50	10145	20343562	0.400
51	10146	20343563	0.400	PYTH3	2017-11-14 09:58:54	52	10147	20343564	0.400
53	10148	20343565	0.400	PYTH3	2017-11-14 09:58:55	54	10149	20343566	0.400
55	10150	20343567	0.400	PYTH3	2017-11-14 09:58:56	56	10151	20343568	0.400
57	10152	20343569	0.400	PYTH3	2017-11-14 09:58:57	58	10153	20343570	0.400
59	10154	20343571	0.400	PYTH3	2017-11-14 09:58:58	60	10155	20343572	0.400
61	10156	20343573	0.400	PYTH3	2017-11-14 09:58:59	62	10157	20343574	0.400
63	10158	20343575	0.400	PYTH3	2017-11-14 09:59:00	64	10159	20343576	0.400
65	10160	20343577	0.400	PYTH3	2017-11-14 09:59:01	66	10161	20343578	0.400
67	10162	20343579	0.400	PYTH3	2017-11-14 09:59:02	68	10163	20343580	0.400
69	10164	20343581	0.400	PYTH3	2017-11-14 09:59:03	70	10165	20343582	0.400
71	10166	20343583	0.400	PYTH3	2017-11-14 09:59:04	72	10167	20343584	0.400
73	10168	20343585	0.400	PYTH3	2017-11-14 09:59:05	74	10169	20343586	0.400
75	10170	20343587	0.400	PYTH3	2017-11-14 09:59:06	76	10171	20343588	0.400
77	10172	20343589	0.400	PYTH3	2017-11-14 09:59:07	78	10173	20343590	0.400
79	10174	20343591	0.400	PYTH3	2017-11-14 09:59:08	80	10175	20343592	0.400
81	10176	20343593	0.400	PYTH3	2017-11-14 09:59:09	82	10177	20343594	0.400
83	10178	20343595	0.400	PYTH3	2017-11-14 09:59:10	84	10179	20343596	0.400
85	10180	20343597	0.400	PYTH3	2017-11-14 09:59:11	86	10181	20343598	0.400
87	10182	20343599	0.400	PYTH3	2017-11-14 09:59:12	88	10183	20343600	0.400
89	10184	20343601	0.400	PYTH3	2017-11-14 09:59:13	90	10185	20343602	0.400
91	10186	20343603	0.400	PYTH3	2017-11-14 09:59:14	92	10187	20343604	0.400
93	10188	20343605	0.400	PYTH3	2017-11-14 09:59:15	94	10189	20343606	0.400
95	10190	20343607	0.400	PYTH3	2017-11-14 09:59:16	96	10191	20343608	0.400
97	10192	20343609	0.400	PYTH3	2017-11-14 09:59:17	98	10193	20343610	0.400
99	10194	20343611	0.400	PYTH3	2017-11-14 09:59:18	100	10195	20343612	0.400
101	10196	20343613	0.400	PYTH3	2017-11-14 09:59:19	102	10197	20343614	0.400
103	10198	20343615	0.400	PYTH3	2017-11-14 09:59:20	104	10199	20343616	0.400
105	10200	20343617	0.400	PYTH3	2017-11-14 09:59:21	106	10201	20343618	0.400
107	10202	20343619	0.400	PYTH3	2017-11-14 09:59:22	108	10203	20343620	0.400
109	10204	20343621	0.400	PYTH3	2017-11-14 09:59:23	110	10205	20343622	0.400
111	10206	20343623	0.400	PYTH3	2017-11-14 09:59:24	112	10207	20343624	0.400
113	10208	20343625	0.400	PYTH3	2017-11-14 09:59:25	114	10209	20343626	0.400
115	10210	20343627	0.400	PYTH3	2017-11-14 09:59:26	116	10211	20343628	0.400
117	10212	20343629	0.400	PYTH3	2017-11-14 09:59:27	118	10213	20343630	0.400
119	10214	20343631	0.400	PYTH3	2017-11-14 09:59:28	120	10215	20343632	0.400
121	10216	20343633	0.400	PYTH3	2017-11-14 09:59:29	122	10217	20343634	0.400
123	10218	20343635	0.400	PYTH3	2017-11-14 09:59:30	124	10219	20343636	0.400
125	10220	20343637	0.400	PYTH3	2017-11-14 09:59:31	126	10221	20343638	0.400
127	10222	20343639	0.400	PYTH3	2017-11-14 09:59:32	128	10223	20343640	0.400
129	10224	20343641	0.400	PYTH3	2017-11-14 09:59:33	130	10225	20343642	0.400
131	10226	20343643	0.400	PYTH3	2017-11-14 09:59:34	132	10227	20343644	0.400
133	10228	20343645	0.400	PYTH3	2017-11-14 09:59:35	134	10229	20343646	0.400
135	10230	20343647	0.400	PYTH3	2017-11-14 09:59:36	136	10231	20343648	0.400
137	10232	20343649	0.400	PYTH3	2017-11-14 09:59:37	138	10233	20343650	0.400
139	10234	20343651	0.400	PYTH3	2017-11-14 09:59:38	140	10235	20343652	0.400
141	10236	20343653	0.400	PYTH3	2017-11-14 09:59:39	142	10237	20343654	0.400
143	10238	20343655	0.400	PYTH3	2017-11-14 09:59:40	144	10239	20343656	0.400
145	10240	20343657	0.400	PYTH3	2017-11-14 09:59:41	146	10241	20343658	0.400
147	10242	20343659	0.400	PYTH3	2017-11-14 09:59:42	148	10243	20343660	0.400
149	10244	20343661	0.400	PYTH3	2017-11-14 09:59:43	150	10245	20343662	0.400
151	10246	20343663	0.400	PYTH3	2017-11-14 09:59:44	152	10247	20343664	0.400
153	10248	20343665	0.400	PYTH3	2017-11-14 09:59:45	154	10249	20343666	0.400
155	10250	20343667	0.400	PYTH3	2017-11-14 09:59:46	156	10251	20343668	0.400
157	10252	20343669	0.400	PYTH3	2017-11-14 09:59:47	158	10253	20343670	0.400
159	10254	20343671	0.400	PYTH3	2017-11-14 09:59:48	160	10255	20343672	0.400
161	10256	20343673	0.400	PYTH3	2017-11-14 09:59:49	162	10257	20343674	0.400
163	10258	20343675	0.400	PYTH3	2017-11-14 09:59:50	164	10259	20343676	0.400
165	10260	20343677	0.400	PYTH3	2017-11-14 09:59:51	166	10261	20343678	0.400
167	10262	20343679	0.400	PYTH3	2017-11-14 09:59:52	168	10263	20343680	0.400
169	10264	20343681	0.400	PYTH3	2017-11-14 09:59:53	170	10265	20343682	0.400
171	10266	20343683	0.400	PYTH3	2017-11-14 09:59:54	172	10267	20343684	0.400
173	10268	20343685	0.400	PYTH3	2017-11-14 09:59:55	174	10269	20343686	0.