SCC 221 – Introdução à Ciência de Computação I

Prova 3 – 03/07/2017

Nome: _____ Nro USP: _____

O Poeta é um fingidor
Finge tão completamente
Que chega a fingir que é dor
A dor que deveras sente.

Fernando Pessoa

Fernando Pessoa é aluno do BCC que, além de computação, gosta muito de poesia. Inclusive escreveu o poema acima! Ele então resolveu combinar poesia com computação e decidiu extrair algumas informações dos poemas que andou escrevendo nas horas vagas.

Um poema é um conjunto de estrofes, sendo cada estrofe, por sua vez, um conjunto de versos (ou sentenças). Todos os poemas têm no máximo 10 estrofes. No entanto, cada estrofe pode ter um conjunto ilimitado de versos e cada verso, por sua vez, também pode ter um conjunto ilimitado de caracteres.

Um poema é formado por letras maiúsculas e minúsculas [a .. z ↔ A .. Z] e os caracteres especiais: '.' '?' '!' ';' e ',' . E nada mais.

Seu programa deverá armazenar para cada poema, a quantidade de estrofes e, para cada estrofe, a quantidade de versos e obviamente os versos . A partir destas informações, você deverá: **1)** imprimir o poema inteiro, relatando a quantidade de estrofes e a quantidade de versos que existem no poema; **2)** Calcular o checksum do poema inteiro; **3)** Calcular o histograma do poema; **4)** Imprimir o poema e na frente de cada verso, imprimir a quantidade de caracteres em cada verso, exceto caracteres especiais.

Para facilitar a captura de informações, a entrada com o poema é formatada da seguinte maneira:

```
1
# 5 \n
Segue o teu destino, \n
Rega as tuas plantas, \n
Ama as tuas rosas. \n
O resto eh a sombra \n
De arvores alheias. \n
# 5 \n
A realidade \n
Sempre eh mais ou menos \n
Do que nos queremos. \n
So nos somos sempre \n
Iguais a nos proprios. \n
* \n
0 \n
```

A entrada ao lado ilustra um caso com apenas um poema (mas poderia haver vários) em que a operação é a de número 1, seguida do poema: O caracter '#' indica uma estrofe, seguido de um valor inteiro (>=1) com a quantidade de versos na mesma. Um poema tem no mínimo uma estrofe. O caracter '*' indica fim de poema. O caracter '0' indica fim da entrada.

Assim, o programa aguarda repetidamente como entrada uma operação (inteiro = 1, 2, 3 ou 4), seguindo de um poema. O programa termina quando a entrada for igual a zero (0).

Cada operação (1,2,3 e 4) será mais bem descrita a seguir. Muito importante: cada operação deverá ser computada posteriormente à leitura do poema (ou seja, não faça a operação solicitada dentro da leitura de todo o poema). Sugestão: implemente cada operação como uma função, utilizando o poema que foi lindo antes. Para isso uma função que lê o poema será fornecida para você. O programa faz uso de locação dinâmica. É sua tarefa também, liberar toda a memória alocada, antes de ler um novo poema.

1) Imprimir: Imprimir o poema substituindo as linhas com os caracteres de controle '#' e '*' por linhas em branco. Acrescida de uma linha extra com o nro de estrofes e o nro de linhas no poema.

2) O checksum é um valor numérico que verifica a integridade de um conjunto de dados. Este checksum é um valor de **8 bits (unsigned)** e consiste em somar todos as letras do poema, **EXCETO os caracteres especiais descrito acima**. Pode acontecer que durante a soma o valor "estoure" (overflow). Este estouro deve simplesmente ser ignorado, continuando a soma normalmente.

3) Histograma do poema: um histograma é definido como um vetor de **ocorrências (acumulador)** das 26 letras maiúsculas do alfabeto [A .. Z]. Note que o poema pode ter letras maiúsculas e minúsculas, que deverão ser acumuladas independemente deste detalhe. Assim, letras minúsculas devem ser convertidas em maiúsculas e incorporadas ao histograma. Os caracteres especiais deverão ser também contabilizados com um só, e colocados na 27ª posição do vetor.

O histograma hist [0..26] consiste de um vetor de 27 posições, sendo hist[0..25] as posições para armazenar a quantidade das letras A .. Z, respectivamente e hist[26] o acumulador para os caracteres especiais.

Portanto, para o poema hipotético abaixo:

aaaaaaaaaabcdefz
AAAAAAAAAA
!!!...????

O histograma teria em hist[0] = 20; hist[1] = hist[2] = hist[3] = hist[4] = hist[5] = hist[25] = 1 e hist[26] = 10 (10 caracteres especias, todos no terceiro verso.

4) Imprimir o poema inteiro (semelhante a operação 1) mas colocando na frente de cada verso o número de caracteres do verso, EXCETO caracteres especiais. Nota sua função deverá ser **recursiva**, não podendo empregar strlen().

Questões:

1. (1 ponto) **Operação = 1**, casos de teste 1 e 2

- **Imprimir Poema**. O programa recebe como entrada um poema formatado como ilustrado na página 1 e deve ler e armazenar todas as estrofes com todos os seus versos.

A seguir, deverá imprimir o poema (sem os caracteres de controle) e ao final uma linha contendo o número de estrofes e o total de linhas do poema inteiro.

Entrada:

```
1
# 4
Quero ignorado, e calmo
Por ignorado, e proprio
Por calmo, encher meus dias
De nao querer mais deles.
# 4
Aos que a riqueza toca
O ouro irrita a pele.
Aos que a fama bafeja
Embacia se a vida.
# 4
Aos que a felicidade
Eh sol, virah a noite.
Mas ao que nada espera
Tudo que vem eh grato.
*
```

Saída:

```
Quero ignorado, e calmo \n
Por ignorado, e próprio \n
Por calmo, encher meus dias \n
De nao querer mais deles. \n

Aos que a riqueza toca \n
O ouro irrita a pele. \n
Aos que a fama bafeja \n
Embacia se a vida. \n

Aos que a felicidade \n
Eh sol, virah a noite. \n
Mas ao que nada espera \n
Tudo que vem eh grato. \n

Este poema tem 3 estrofes e 12 versos \n
```

- Atençao, antes de ler novamente um novo poema, você deverá liberar toda a memória que foi previamente alocada em uma outra operação. Observe a função função vazia **apaguePoema()** em **lePoema()**. Personalize apaguePoema().

2. (3 pontos) **Operação = 2**, casos de teste 3 e 4

- **CheckSum**: o checksum é a somatória de todos os caracteres do poema (na realidade o valor ASCII de cada caracter), EXCETO os caracteres especiais, em uma variável de 8 bits. Conforme descrito anteriormente, se houver estouro (overflow) este deve ser desprezado.

Entrada:

```
2
# 1
aA
# 1
.!?
# 1
bc
*
```

Saída:

```
O CheckSum deste poema eh = 103 \n
```

3. (3 pontos). **Operacao = 3**, casos de teste 5 e 6.

- **Histograma do poema**. Veja descrição sobre histograma na página 2.

Entrada:

```
3
# 3
aaaaaaaaaabcdefz
AAAAAAAAAA
!!!...????
*
```

Saída:

```
A   B   C   D   E   F   G   H   I   J   K   L   M   N   O   P   Q   R   S   T   U   V   W   X   Y   Z \n
20  1   1   1   1   1   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   1   10 \n
```

- **Atençao**: a saída é composta de 2 linhas. Na primeira linha são impressas todas as 26 letras maiusculas, seguidas de ‘\n’. Na segunda linha, estão os valores correspondentes para as 26 letras [A .. Z] e o valor para os caracteres acumulados. Cada letra e valor ocupam um espaço horizontal equivalente a 4 caracteres, tabulado à esquerda, ou seja → printf (“%-4”)

4. (3 pontos). **Operacao = 4**, casos de teste 7 e 8

- **Imprimir o poema** e na frente colocar a **quantidade de caractes**, NÃO contabilizando os caracteres especiais ‘.’ ‘?’ ‘!’ ‘;’ e ‘,’ . Esta função deverá ser recursiva e não pode conter strlen().

Entrada:

```
3
# 6
Oh mar salgado, quanto do teu sal
Sao lagrimas de Portugal!
Por te cruzarmos, quantas maes choraram,
Quantos filhos em vao rezaram!
Quantas noivas ficaram por casar
Para que fosses nosso, o mar!
# 6
Valeu a pena? Tudo vale a pena
Se a alma nao eh pequena.
Quem quer passar alem do Bojador
Tem que passar alem da dor.
Deus ao mar o perigo e o abismo deu,
Mas nele eh que espelhou o ceu.
*
```

Saída:

```
Oh mar salgado, quanto do teu sal -> 32 \n
Sao lagrimas de Portugal! -> 24 \n
Por te cruzarmos, quantas maes choraram, -> 38 \n
Quantos filhos em vao rezaram! -> 29 \n
Quantas noivas ficaram por casar -> 32 \n
Para que fosses nosso, o mar! -> 27 \n

Valeu a pena? Tudo vale a pena -> 29 \n
Se a alma nao eh pequena. -> 24 \n
Quem quer passar alem do Bojador -> 32 \n
Tem que passar alem da dor. -> 26 \n
Deus ao mar o perigo e o abismo deu, -> 35 \n
Mas nele eh que espelhou o ceu. -> 30 \n
```

Total de pontos na prova : 1 + 3 + 3 + 3 = 10
1 ponto extra para quem implementar a função apaguePoema()

Portanto, a prova vale 11 pontos.

Você muito provavelmente vai querer saber qual é o valor ASCII de alguns caracteres...

Tabela ASCII para consulta:

ASCII printable characters

32	space	64	@	96	`
33	!	65	A	97	a
34	"	66	B	98	b
35	#	67	C	99	c
36	$	68	D	100	d
37	%	69	E	101	e
38	&	70	F	102	f
39	'	71	G	103	g
40	(	72	H	104	h
41	)	73	I	105	i
42	*	74	J	106	j
43	+	75	K	107	k
44	,	76	L	108	l
45	-	77	M	109	m
46	.	78	N	110	n
47	/	79	O	111	o
48	0	80	P	112	p
49	1	81	Q	113	q
50	2	82	R	114	r
51	3	83	S	115	s
52	4	84	T	116	t
53	5	85	U	117	u
54	6	86	V	118	v
55	7	87	W	119	w
56	8	88	X	120	x
57	9	89	Y	121	y
58	:	90	Z	122	z
59	;	91	[	123	{
60	<	92	\	124	|
61	=	93	]	125	}
62	>	94	^	126	~
63	?	95	_		

Boa prova a todos.

Sei Que Nunca Terei o Que Procuro

Sei que nunca terei o que procuro
E que nem sei buscar o que desejo,
Mas busco, insciente, no silêncio escuro
E pasmo do que sei que não almejo.

Fernando Pessoa.