

AED2 2021 - TAREFA 6 - BRINCANDO COM AS PALAVRAS ORDENADAS

Entrega: 26/05/2021 até 23:59:59

Instruções:

1. E/S: tanto a entrada quanto a saída de dados devem ser "secas", ou seja, não devem apresentar frases explicativas. Siga o modelo fornecido e apenas complete as partes informadas (veja o exemplo abaixo).
2. Identificadores de variáveis: escolha nomes apropriados
3. Documentação: inclua cabeçalho, comentários e indentação no programa.
4. Submeta o programa no sistema judge utilizando acesso remoto via VPN: `http://judge.sjc.unifesp.br/aed2`, ou através de conexão direta: `http://kp.unifesp.br:9001/aed2/login`.
5. O código-fonte pode ser escrito em C, C++ ou Java.
6. **O código-fonte DEVE implementar uma solução usando o algoritmo heapsort da aula gravada (Prof. Álvaro) e dos slides disponibilizados**

Descrição:

Encontrar formas de trabalhar com palavras ou ordenação sempre foram problemas importantes na ciência da computação. Sabendo disso, um grupo de amigos resolveu criar uma brincadeira: "*Brincando com as palavras ordenadas*". Cada participante terá um papel no jogo. O jogo começa com um dos participantes sorteando um número inteiro positivo N que representa a quantidade de palavras, depois o segundo participante escreve as N ($N < 1000$) palavras em um papel sem qualquer critério, ou seja, usando letras, números, caracteres especiais, etc. O terceiro especifica uma quantidade M ($M < N$) de palavras a serem escolhidas do conjunto original, e o quarto jogador seleciona M posições distintas no intervalo $[0, N[$ referente as palavras a serem selecionadas. O último jogador deverá ter a função de ordenar um conjunto de M palavras selecionadas, usando para isso o algoritmo de ordenação *Heap-Sort*, o qual também deverá comprovar exibindo o *heap* montado a partir do conjunto selecionado. Porém, o jogo tem algumas restrições:

1. Só palavras minúsculas sem acentuação podem ser aceitas. Caso alguma tenha algum caractere diferente, a primeira palavra que não segue essa regra deve ser impressa como: *A palavra XXX eh invalida*.
2. A quantidade máxima de palavras é menor que 1000.
3. Cada palavra tem no máximo 20 caracteres.
4. É **requisito** usar o algoritmo *Heap-Sort* para ordenar e nenhum outro pode ser usado, senão será zerado não só no jogo mas também na nota!(ôlô)

Exemplo:

- O primeiro jogador especifica 7 como o número de palavras do conjunto original de entrada.
- O segundo jogador escreve as palavras desse conjunto: "**programar vamos palavra eh futebol computador legal**".
- O terceiro jogador informa a quantidade de palavras para formar o conjunto resultante: 3.
- O quarto jogador informa os índices das palavras do conjunto original que irão compor o conjunto resultante: 0, 3 e 6.
- O quinto jogador ordena o conjunto resultante e escreve o *heap* criado e na linha seguinte o conjunto ordenado das palavras selecionadas: "**eh legal programar**".

Importante notar que apenas o novo conjunto (vetor) com as palavras selecionadas deve ser ordenado.

ENTRADA:

Primeira linha contém a quantidade N do total de palavras do conjunto inicial.

Na segunda linha contém as N palavras separadas por um espaço em branco, representando o conjunto inicial de palavras: p_1, p_2, \dots, p_N .

Na terceira linha refere-se a quantidade (M) de palavras para formar o conjunto de palavras selecionadas.

A quarta linha contém uma sequência de inteiros que representam as posições ou índices ($0 : N - 1$) separados por um espaço em branco, representando a sequência de índices das palavras a serem selecionados que formarão o novo conjunto (o qual deverá ser ordenado).

SAÍDA:

Caso alguma palavra do conjunto de entrada de N palavras não esteja de acordo com a regra, a **primeira** palavra fora do padrão deve ser impressa conforme os exemplos exibidos a seguir. Caso todas as palavras do conjunto de entrada estejam de acordo com as regras, a primeira linha da saída representa o conjunto *heap* de tamanho M , criado pela rotina *build_heap*, de acordo com o mostrado na videoaula a respeito. A forma de exibir essa informação segue o padrão dado pelos exemplos a seguir, onde se exibe a *string*: "build_heap:" seguido do próprio vetor *heap* construído. A segunda linha da saída é composta pelo pela *string* "palavras: " e o novo conjunto ordenado das M palavras selecionadas, onde cada palavra exibida é **separada** por um espaço em branco. Importante lembrar que a última palavra **não** deve ter um espaço após a mesma.

Exemplos de entrada e saída:

Exemplos de entrada	Exemplos de saída
7 programar vamos palavra eh futebol computador legal 3 0 3 6	build_heap: programar eh legal palavras: eh legal programar
6 banana tomate amora laranja limao jabuticaba 3 1 2 5	build_heap: tomate amora jabuticaba palavras: amora jabuticaba tomate
5 todos estes que aí estao 2 0 1	a palavra aí eh invalida
4 ver uma Estrela Cadente 2 0 1	a palavra Estrela eh invalida

Tabela 1: Exemplos de entrada e saída