

Corrida das capivaras

Trabalho 1

Algoritmos e Programação II

1 Descrição

SAPIVARAS são bichinhos muito *kawaii*. Achadas em certas áreas das Américas do Sul e Central, próximo a rios e lagos, a capivara (*Hydrochoerus hydrochaeris*), também chamada de carpincho e capincho, é o maior roedor do mundo. Quando a esquadra de Pedro Álvares Cabral chegou ao Brasil em 1500, os indígenas locais já domesticavam este animal. Alimenta-se de capins e ervas, daí, a etimologia de seu nome: capivara procede do termo tupi *kapi'wara*, que significa “comedor de capim”. Já capincho vem do castelhano platino *capincho*. No Rio Grande do Sul, é também conhecida por capinga.

A vida das capivaras na UFMS não é fácil. Com uma agenda sempre lotada, suas diversas atividades incluem passeios *fitness* na FACOM, natação e hidroginástica, e tarefas de Algoritmos e Programação I. Uma das atividades preferidas das capivaras é a corrida das capivaras.



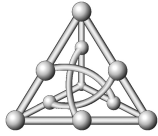
Tem como não amar?



Capivara na velocidade da luz

A corrida das capivaras é uma competição com regras um pouco diferentes do habitual. Nela, n capivaras são posicionadas em fila, a 1ª ficando logo após a linha de largada, e a última (n -ésima) ficando após todas capivaras. Além disso, cada uma delas recebe um número de 1 a n de acordo com sua posição na fila. A capivara ao lado recebeu o número 1, o que significa que ela é a primeira da fila.

Nessa competição, a cada ultrapassagem que uma capivara faz, ela ganha **um ponto**. Ao final da corrida, a capivara vencedora é aquela que tiver ganho **mais pontos**, **não** a que chegar primeiro ao fim. Caso duas ou mais capivaras terminem a corrida com a mesma quantidade de pontos, o desempate é feito pela posição inicial na fila: aquela que tem uma numeração menor ganha, já que uma capivara que começa à frente tem menos adversários para ultrapassar. Ao final da corrida, queremos saber a ordem em que as capivaras cruzaram a linha de chegada e a classificação de todas capivaras, da primeira colocada para a última, usando o esquema de pontuação e desempate descrito acima. Esse é seu trabalho.



2 Entrada e saída

A **entrada**¹ contém um caso de teste representando um evento de corrida. A primeira linha contém um número inteiro $n > 0$ que se refere à quantidade de capivaras participantes. Implicitamente, sabemos que as capivaras serão colocadas em fila e cada uma receberá um número de acordo com sua posição na largada, de 1 a n . A seguir, são apresentadas as ultrapassagens, uma por linha, que deverão ser lidas até seu programa encontrar o fim de arquivo (EOF). Uma ultrapassagem corresponde a um único número, e representa que a capivara com esse número ultrapassou aquela que estava à sua frente naquele instante. Observe que a capivara que estiver na primeira posição em algum instante da corrida nunca fará uma ultrapassagem, a menos que outra capivara a ultrapasse primeiro.

Como **saída**², seu programa deve escrever em uma linha a ordem de chegada das capivaras (da primeira a chegar para a última a chegar) separadas por espaço, e na linha seguinte a ordem de classificação das capivaras (da melhor colocada para a pior colocada) separadas por espaço, seguindo o critério de pontuação e desempate descrito anteriormente.

Exemplo de entrada 1

```
5
4
4
2
```

Exemplo de saída 1

```
1 2 4 3 5
4 2 1 3 5
```

Exemplo de entrada 2

```
4
3
2
```

Exemplo de saída 2

```
1 2 3 4
2 3 1 4
```

Exemplo de entrada 3

```
10
8
4
5
3
3
2
3
9
7
```

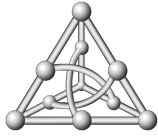
Exemplo de saída 3

```
2 3 1 4 5 6 8 7 9 10
3 2 4 5 7 8 9 1 6 10
```

(continua na próxima página... ↓)

¹A entrada consiste apenas naquilo que o programa lê, utilizando *scanf* e similares

²A saída consiste apenas naquilo que o programa escreve, utilizando *printf* e similares



3 Exigências

Você **DEVE** usar a seguinte estrutura de dados em seu trabalho:

```
/* Armazena informações de uma capivara */
typedef struct {
    int numero;           /* Número da capivara = posição na largada */
    int ultrapass;        /* Quantidade de ultrapassagens feitas */
} capivara;
```

Esse registro (um vetor deles) já é suficiente para armazenar as informações necessárias sobre as capivaras e resolver o problema, mas se achar necessário você pode adicionar outros campos ao registro. Não é permitido o uso de outras estruturas ou vetores.

Além disso, como você pôde perceber pelo enunciado do trabalho, em alguns pontos há necessidade de ordenar um conjunto de dados. Você **DEVE** usar um método de ordenação não-elementar para realizar essas tarefas, tais como ordenação por intercalação ou ordenação por separação, **implementado por você**.

Caso alguma exigência não seja cumprida, a nota é **ZERO**.

4 Entrega

Instruções para entrega do seu trabalho:

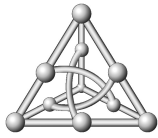
1. Cabeçalho

Seu trabalho deve ter um cabeçalho com o seguinte formato:

```
/*
 *
 * Nome do(a) estudante
 * Trabalho 1
 * Professor(a): Nome do(a) professor(a)
 *
 */
```

2. Compilador

Os(as) professores(as) usam o compilador da linguagem C da coleção de compiladores GNU `gcc`, com as opções de compilação `-Wall -std=c99 -pedantic` para corrigir os programas. Se você usar algum outro compilador para desenvolver seu programa, antes de entregá-lo verifique se o seu programa tem extensão `.c`, compila sem mensagens de alerta e executa corretamente.



3. Forma de entrega

A entrega será realizada diretamente na página da disciplina no [AVA/UFMS](#). Um fórum de discussão deste trabalho já se encontra aberto. Após abrir uma sessão digitando seu *login* e sua senha, vá até o tópico “Trabalhos”, e escolha “T1 - Entrega”. Você pode fazer o upload de vários rascunhos, mas **o envio definitivo deve ser feito até a data indicada no AVA**. Apenas com o envio definitivo seu trabalho será corrigido. Encerrado o prazo, não serão mais aceitos trabalhos.

4. Atrasos

Trabalhos atrasados não serão aceitos. Não deixe para entregar seu trabalho na última hora. Para prevenir imprevistos como queda de energia, problemas com o sistema, falha de conexão com a internet, sugerimos que a entrega do trabalho seja feita pelo menos um dia antes do prazo determinado.

5. Erros

Trabalhos com erros de compilação receberão nota **ZERO**. Faça todos os testes necessários para garantir que seu programa está livre de erros de compilação.

6. O que entregar?

Você deve entregar um único arquivo contendo **APENAS** o seu programa fonte com o mesmo nome de seu login no passaporte UFMS, como por exemplo, `fulano.silva.c`. **NÃO** entregue qualquer outro arquivo, tal como o programa executável, já compilado.

7. Verificação dos dados de entrada

Não se preocupe com a verificação dos dados de entrada do seu programa. Seu programa não precisa fazer consistência dos dados de entrada. Isto significa que se, por exemplo, o seu programa pede um número entre 1 e 10 e o usuário digita um número negativo, uma letra, um cifrão, etc, o seu programa pode fazer qualquer coisa, como travar o computador ou encerrar a sua execução abruptamente com respostas erradas.

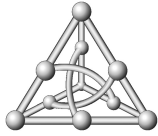
8. Arquivo com o programa fonte

Seu arquivo contendo o programa fonte na linguagem C deve estar bem organizado. Um programa na linguagem C tem de ser muito bem compreendido por uma pessoa. Verifique se seu programa tem a indentação adequada, se não tem linhas muito longas, se tem variáveis com nomes significativos, entre outros. Não esqueça que um programa bem descrito e bem organizado é a chave de seu sucesso. Não esqueça da documentação de seu programa e de suas funções.

Dê o nome do seu usuário do Passaporte UFMS para seu programa e adicione a extensão `.c` a este arquivo. Por exemplo, `fulano.silva.c` é um nome válido.

9. Conduta Ética

O trabalho deve ser feito **INDIVIDUALMENTE**. Cada estudante tem responsabilidade sobre cópias de seu trabalho, mesmo que parciais. Não faça o trabalho em grupo e não compartilhe seu programa ou trechos de seu programa. Você pode consultar seus colegas para



esclarecer dúvidas e discutir idéias sobre o trabalho, ao vivo ou no fórum de discussão da disciplina, mas **NÃO** copie o programa!

Trabalhos envolvidos em plágio, mesmo que parcial, terão nota **ZERO**.