Algoritmos e Estruturas de Dados 1

Segundo Trabalho Prático (Primeira Versão)

Entrega Sugerida: 12/02/2022

2º Semestre 2022 - DC-UFSCar

1 Introdução

No segundo Trabalho Prático (TP02) será solicitada a entrega de um programa que solucione o problema apresentado na próxima seção. O arquivo entregue deve seguir os itens abaixo:

- O TP02 deverá ser feito individualmente e plágio não será tolerado;
- O TP02 deve ser entregue no run codes (https://run.codes) em um arquivo contendo código em linguagem C, e com um cabeçalho com as informações do estudante (nome, curso, RA);
- Cada estudante deve se cadastrar no run codes (https://run.codes) informando Nome Completo, escolhendo "UFSCar - Universidade Federal de São Carlos" no campo Universidade e colocando seu RA no campo Núm. Matrícula. Depois de cadastrado, basta logar no run codes e se matricular na disciplina "1001502 - Algoritmos e Estruturas de Dados 1" usando o Código de Matrícula ZH6B.
- Compile o seu TP usando o compilador GCC com flags -Wall -pedantic -O2 -Wno-unused-result, pois warnings podem impedir o código de funcionar no run codes ainda que funcionem no seu computador;

2 Problema de Josephus

Conta-se uma história sobre o fim de uma guerra hebraica, na qual o historiador Josephus e mais 40 homens foram cercados pelo exército romano. Eles decidiram não se entregar com vida, mas sua religião condena o suicídio. Por isso, fizeram um pacto no qual os 41 homens fariam um círculo e, começando pelo homem na primeira posição, um homem de cada vez mataria aquele imediatamente à sua esquerda, sendo que o próximo homem a matar é o que estava a esquerda do último a morrer. Nesse contexto, Josephus precisou encontrar em qual lugar no círculo deveria se posicionar para que fosse o último homem que restasse. Essa história dá origem ao **Problema de Josephus** que será o exercício a ser resolvido nesse trabalho. O vídeo "The Josephus Problem - Numberphile" é bastante interessante e explora mais a fundo este problema - `https://goo.gl/MVNcB6`.

3 Tarefas

Crie um arquivo trabalho3.c contendo um cabeçalho com as informações do estudante, uma função *main()* e uma função *int resolveJosephus(int n, int m)*, seguindo as instruções das seguintes subseções.

3.1 Função: main()

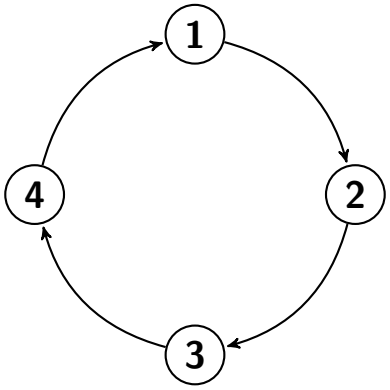
Sua função main deve ler o numero de execuções do problema de Josephus e posteriormente os inteiros m, representando o número de pessoas, e p, representando o tamanho do passo, e devolver o resultado de cada execução. Para isso, deve-se usar a seguinte implementação do programa main.

```
1  int main (){
2      int nroexecs;
3
4      scanf("%d", &nroexecs);
5      int *n = malloc(nroexecs * sizeof(int));
6      int *m = malloc(nroexecs * sizeof(int));
7
8      for(int i = 0; i < nroexecs; i++){
9          scanf("%d", &n[i]);
10         scanf("%d", &m[i]);
11     }
12
13     for(int i = 0;i<nroexecs; i++){
14         printf("Usando n=%d, m=%d, resultado=%d\n", n[i], m[
       i], resolveJosephus(n[i], m[i]));
15     }
16
17     return 0;
18 }
```

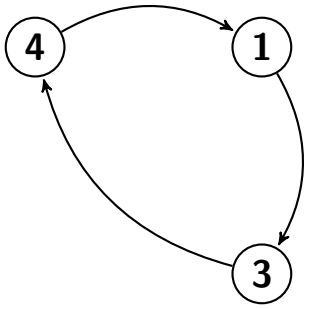
3.2 Função resolveJosephus(int n, int m)

Suponha que n pessoas estão dispostas em um círculo e que são numeradas de 1 a n no sentido horário. Comece pela pessoa de número 1 e elimine a (m+1)-ésima pessoa. Em seguida, contando a partir da pessoa p que sucede a última eliminada, elimine a pessoa que está m posições depois de p. Repita este processo enquanto o circulo tiver duas ou mais pessoas. Note que, como estamos falando de um círculo, ao andar m posições a partir de uma pessoa, podemos acabar chegando em pessoas que estão antes dela, ou até mesmo a ela própria. Como a pessoa nunca pode matar a si própria, neste último caso ela deve matar quem estiver imediatamente à frente dela.

Segue um exemplo da resolução do problema com $n = 4$ e $m = 1$:



A primeira pessoa a ser eliminada é a que se encontra na posição 2.



A próxima pessoa eliminada se encontra na posição 4, já que 3 sucede o último eliminado e $m = 1$ posições depois de 3 está o 4.

A última pessoa eliminada se encontra na posição 3 e a sobrevivente é a que se encontra na posição 1.

A função *int resolveJosephus(int n, int m)* deve receber como parâmetros os

inteiros n, que representa o número de pessoas no círculo, e m, que representa o tamanho do passo (número de pessoas puladas no círculo) para a eliminação do próximo. Ela deve devolver a posição do sobrevivente.

3.3 Exemplo de teste

Entrada:

```
3
10
1
10
2
10
3
```

Saída:

```
Usando n=10, m=1, resultado=5
Usando n=10, m=2, resultado=10
Usando n=10, m=3, resultado=6
```

4 Referências interessantes

Josephus Problem - Wikipedia: `https://en.wikipedia.org/wiki/Josephus_problem`

The Josephus Problem - Numberphile (vídeo) `https://goo.gl/MVNcB6`

Site Projeto de Algoritmos: `https://www.ime.usp.br/~pf/algoritmos/`