SCC-502 Algoritmos e Estruturas de Dados 1 **M. Cristina/Jorge**

Trabalho 1 – Torres de Hanói

O trabalho deverá ser feito **individualmente ou em duplas**
e enviado para o e-mail "jorgehpopae@gmail.com" com o assunto
"TRABALHO_1 - <PRIMEIRONOME>_<NºUSP>", por exemplo,
"TRABALHO_1 – Jorge_9999999".

Os nomes das duplas devem ser informados por e-mail até **30/08.**
Assunto: "DUPLA_TRABALHO1"
Corpo do e-mail:
"<Nome Integrante 1>. <Nº USP>
<Nome Integrante 2>. <Nº USP>"

O e-mail deve conter no anexo apenas 1 arquivo compactado,
"TRABALHO1_<NOME1>_<NºUSP1>_<NOME2>_<NºUSP2>.zip", por exemplo,
"TRABALHO1_Jorge_9999999_Henrique_8888888.zip".

Esse arquivo deve conter:

- O código fonte do trabalho;
- O arquivo "makefile" para compilar o trabalho. O arquivo resultante da compilação deve ser um arquivo executável **"trab1.exe"** (tanto para usuários Windows como Linux)

Avaliação:

- 0,5 – Corretude do programa e do TAD (o programa deve fazer o que foi especificado);
- 0,3 – Limpeza e organização do código (código comentado e identado);
- 0,2 – Modularização (TAD Pilha e programa principal em arquivos separados, divisão do código em sub-rotinas).

Aviso 1: Informações de como criar um makefile estão na página do material didático da disciplina no link: <http://wiki.icmc.usp.br/images/6/63/Ponti_Makefile.pdf>

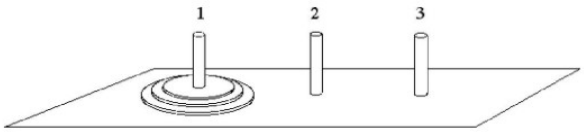
Aviso 2: Atenção para o formato de entrada e saída de dados, especificados neste documento.

Aviso 3: Dia **30/08** ocorrerá um plantão de dúvidas do trabalho, na sala 1-007 (VICG).

A data limite para entrega do trabalho é **20/09**.

Descrição do Trabalho

Torres de Hanói é um quebra-cabeça muito antigo e conhecido. Ele é constituído de um conjunto de N discos de tamanhos diferentes e três pinos verticais, nos quais os discos podem ser encaixados.



Cada pino pode conter uma pilha com qualquer número de discos, **desde que cada disco não seja colocado acima de outro disco de menor tamanho**. A configuração inicial consiste de todos os discos no pino 1. O objetivo é mover todos os discos para o pino 3, sempre obedecendo à restrição de não colocar um disco sobre outro menor.

Escreva uma aplicação que implemente este jogo **utilizando pilhas**.

Esta aplicação deve:

a) Inicializar os pinos (o primeiro deve conter N discos, os demais ficam vazios);
b) Fornecer o método void realizaJogada(pilha *orig, pilha *dest) que faz o movimento de um disco de uma pilha para outra, caso a jogada seja legal. Se a jogada for ilegal, o movimento não é realizado;
c) Informar que o jogador ganhou o jogo quando todos os discos tiverem passado para o pino 3, ou que o jogador perdeu se sobrar algum disco fora do pino 3 depois das jogadas.
d) Cumprir as especificações de **Entrada** e **Saída**, descritas abaixo.

Entrada

A entrada deve ser lida de um arquivo de texto, informado na chamada do programa.

Exemplo (veja um programa exemplo no final do documento):

C:\> programa.exe entrada.dat

A entrada contém vários casos de teste.

A primeira linha de cada caso de teste contém um inteiro N representando o número de discos (3<=N<=15).

A segunda linha de cada caso de teste contém um inteiro J representando o número de jogadas.

As J próximas linhas contém 2 números, separados por espaço. O primeiro é o pino de origem e o segundo é o pino de destino. Lembre-se: **se a jogada for inválida, o movimento não é realizado**.

Saída

A saída deve ser apresentada na saída padrão, através do comando **printf**

Para cada caso de teste, seu programa deve escrever uma linha, contendo um inteiro:

- **JV** se o jogador ganhou o jogo nas J jogadas (todos os discos estão no pino 3). **JV** é o **Número de Jogadas Válidas.**
- **0** se o jogador perdeu o jogo nas J jogadas (algum disco não está no pino 3).

Exemplo de execução

Entrada	Saída
3 7 1 3 1 2 3 2 1 3 2 1 2 3 1 3	7
4 16 1 2 1 3 2 3 1 2 3 1 3 2 1 2 1 3 2 3 2 1 3 1 2 3 1 2 1 3 2 3 1 3	15

3 5 1 3 2 3 2 1 2 3 1 3	0
3 7 1 2 1 3 1 3 3 2 2 1 2 3 1 3	0

Exemplo de Passagem de parâmetros para o programa principal (main)

```
#include <stdio.h>


int main(int argc, char *argv[]) {
	int cont;
	for (cont = 0; cont < argc; cont++){
		printf("Parametro %d: %s\n",cont,argv[cont]);
	}
	return 1;
}
```

Compilação:
C:\> gcc programa.c programa.exe

Execução:
C:\> programa.exe Parametro1 Parametro2 ParametroFinal

Saída:
Parametro 0: programa.exe
Parametro 1: Parametro1
Parametro 2: Parametro2
Parametro 3: ParametroFinal