TRABALHO ACADÊMICO – AV3

EXERCÍCIO 1

Escreva um programa em C que inverta o conteúdo de uma pilha usando uma fila. As instruções detalhadas são as seguintes:

1 - descompacte e abra no Code Blocks o projeto `exemplo_2` disponível no AVA, no *card* MIDIATECA DA DISCIPLINA, seção "Listas, Pilhas e Filas", arquivo "Implementações - Listas, Pilhas e Filas.rar";

2 - modifique o arquivo `main.c` para executar as seguintes instruções:

2.1 - solicite ao usuário que informe um conjunto de valores inteiros;

2.2 - imprima os valores informados;

2.3 - insira os valores informados em uma pilha;

2.4 - imprima o conteúdo da pilha;

2.5 - implemente, no próprio `main.c`, a função que faça a inversão da pilha usando uma fila, cuja assinatura é:

```
void inverte(TipoPilha *p)
```

2.6 - execute a função `inverte( );`

2.7 - imprima o conteúdo da pilha.

Após a execução do programa, copie todo o conteúdo exibido na tela em um arquivo texto com o nome `resultados.txt`.

EXERCÍCIO 2

Escreva um programa em C que, incialmente, crie e insira elementos em duas filas F_1 e F_2. Em seguida, crie uma fila F_3 e transfira alternadamente os elementos de F_1 e F_2 para F_3. As instruções detalhadas são as seguintes:

1 - descompacte e abra no Code Blocks o projeto `exemplo_2` disponível no AVA, no *card* MIDIATECA DA DISCIPLINA, seção "Listas, Pilhas e Filas", arquivo "Implementações - Listas, Pilhas e Filas.rar";

2 - modifique o arquivo `main.c` para executar as seguintes instruções:

2.1 - solicite ao usuário que informe os valores a serem inseridos em F_1 e F_2 e faça as devidas inserções;

2.2 - imprima F_1 e F_2;

2.3 - implemente, no próprio `main.c`, a função que faça a transferência dos elementos de F_1 e F_2 para F_3, cuja assinatura é:

```
TipoFila *transfere(TipoFila *f1, TipoFila *f2)
```

2.4 - execute a função `transfere( )`;

2.5 - imprima F_3.

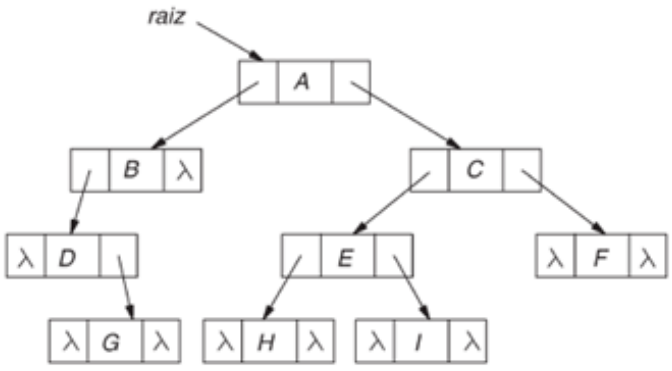
Após a execução do programa, copie todo o conteúdo exibido na tela no mesmo arquivo `resultados.txt` do exercício anterior.

EXERCÍCIO 3

Escreva um programa em C que percorra e imprima o conteúdo dos nós de uma árvore binária. As instruções detalhadas são as seguintes:

1 - descompacte e abra no Code Blocks o projeto `exemplo_1` disponível no AVA, no *card* MIDIATECA DA DISCIPLINA, seção "Árvores Binárias", arquivo "Implementação - Árvores Binárias.rar";

2 - altere o exemplo de árvore binária existente no arquivo `main.c` para que represente a árvore binária apresentada a seguir:



3 - imprima árvore;

4 - implemente, no próprio arquivo `main.c`, as funções a seguir:

```
percorreA(raiz)
   percorreB(subárvore esquerda);
   percorreA(subárvore direita);
   escreva(raiz);
```

```
percorreB(raiz)
   escreva(raiz);
   percorreA(subárvore esquerda);
   percorreB(subárvore direita);
```

4 - execute a função `percorreA( )` a partir da raiz da árvore.

Após a execução do programa, copie todo o conteúdo exibido na tela no mesmo arquivo `resultados.txt` do exercício anterior.

EXERCÍCIO 4

Escreva um programa em C que construa uma árvore binária de busca a partir de um conjunto de números aleatórios inteiros. Em seguida, construa uma nova árvore binária de busca com as chaves ímpares removidas da primeira árvore. As instruções detalhadas são as seguintes:

1 - descompacte e abra no Code Blocks o projeto `exemplo_2` disponível no AVA, no *card* MIDIATECA DA DISCIPLINA, seção “Árvores Binárias de Busca”, arquivo “Implementação - Árvores Binárias de Busca.rar”;

2 - modifique o arquivo `main.c` para executar as seguintes instruções:

2.1 - solicite ao usuário que informe a quantidade de números aleatórios inteiros a ser gerada;

2.2 - gere a sequência de números aleatórios;

2.3 - imprima a sequência gerada;

2.4 - construa a árvore binária de busca usando os números da sequência como chaves dos nós;

2.5 - imprima a árvore;

2.6 - construa uma nova árvore binária de busca com os nós de chave ímpar removidos da árvore inicial;

2.7 - imprima a nova árvore.

Após a execução do programa, copie todo o conteúdo exibido na tela no mesmo arquivo `resultados.txt` do exercício anterior.

OBSERVAÇÕES SOBRE A ENTREGA E A AVALIAÇÃO:

- Os projetos com os TAD's fornecidos para os exercícios devem ser obrigatoriamente utilizados;

- Devem ser postados no AVA apenas o arquivo `main.c` de cada exercício e o arquivo `resultados.txt`. O arquivo do exercício 1 deve ser nomeado como `main_1.c`, o do exercício 2 deve ser nomeado como `main_2.c` e assim sucessivamente.

- Para cada exercício, o arquivo `main.c` correspondente será testado no projeto original, por isso, não se deve fazer alterações em quaisquer outros arquivos do projeto para não interferir no funcionamento do programa.

- Caso o arquivo `main.c` de um exercício não funcione no projeto original correspondente, a nota desse exercício será **0,0**.

- A postagem dos arquivos deve acontecer, **impreterivelmente**, até o dia **02/12/20** às **23:55**.