Lista de exercícios 2

1. Considerando a entrada e a sequência de operações (inserir e retirar) em uma pilha, mostre a saída resultante.

a)

Entrada: I E N S R E

Sequência: I R I I R I R I I R R R

Saída (elementos retirados da pilha, preservar a ordem):

5	
4	
3	
2	
1	
0	

b)

Entrada: A, B, C, D, E, F

Sequência: I I R I I R I R R I R R

Saída (elementos retirados da pilha, preservar a ordem):

5	
4	
3	
2	
1	
0	

2. Considere um conjunto de 4 números inteiros na seguinte ordem: 7, 20, 6 e 15. Qual seria a sequência correta de operações de inserção (I) e retirada (R) em uma pilha para se obter os registros na ordem 20 15 6 7?

Sequência de operações: _____

3. Resolva a questão abaixo:

A pilha é uma estrutura de dados que permite a inserção/ remoção de itens dinamicamente seguindo a norma de último a entrar, primeiro a sair. Suponha que para uma estrutura de dados, tipo pilha, são definidos os comandos:

- PUSH (p, n): Empilha um número "n" em uma estrutura de dados do tipo pilha "p";
- POP (p): Desempilha o elemento no topo da pilha.

Considere que, em uma estrutura de dados tipo pilha "p", inicialmente vazia, sejam executados os seguintes comandos:

```
PUSH (p, 10)
PUSH (p, 5)
PUSH (p, 3)
PUSH (p, 40)
POP (p)
PUSH (p, 11)
PUSH (p, 4)
PUSH (p, 7)
POP (p)
POP (p)
```

Após a execução dos comandos, o elemento no topo da pilha "p" e a soma dos elementos armazenados na pilha "p" são, respectivamente,

A) 11 e 29.
B) 11 e 80.
C) 4 e 80.
D) 7 e 29.
E) 7 e 40.

4. Implemente um programa que manipule uma pilha. O programa deve ser desenvolvido com as seguintes regras:
    - Criar um menu com as opções:
        - Criar pilhar – O usuário deve informar o tamanho da pilha; caso a pilha já tenha sido criada anteriormente, antes de criar uma pilha nova, a pilha antiga deve ser removida e a memória deve ser desalocada.
        - Inserir – O usuário deve informar um valor a ser inserido; caso a pilha esteja cheia, uma mensagem deve ser exibida.
        - Remover – Remover um elemento da pilha e mostrar o elemento removido; caso a pilha esteja vazia, uma mensagem deve ser exibida.
        - Consultar – Verificar se a pilha contém um determinado valor informado pelo usuário.
        - Mostrar – Mostar a pilha.
        - Sair – Sair do programa.
    - Não permitir a inclusão de valores duplicados;
    - Imprimir uma mensagem de erro caso a pilha não tenha sido criada e o usuário tente executar uma das seguintes operações: inserir, remover, consultar ou mostrar.
5. Faça um programa que cadastre em uma estrutura do tipo pilha vários números, sendo no máximo 6, ao remover um número desta estrutura (desempilhado) o mesmo deve ser empilhado em outra pilha, conforme o critério: se o número for par na pilha dos pares, se for ímpar na pilha dos ímpares. No menu de opções deve ter uma alternativa para zerar as três pilhas, mostrar pilha inicial, a dos pares e a dos ímpares.

6. Crie uma função para buscar um valor **v** em uma pilha **P** fazendo uso de uma pilha auxiliar **AUX**. Observações:
   - Protótipo da função:
     `bool buscaValor(Pilha *p, int valor);`
   - Não é permitido o uso de vetores auxiliares.
   - Após a chamada da função, a pilha **P** deve conter os mesmos elementos, os valores devem ser preservados.
   - Implementação da função:
     - Desempilhar os elementos de **P** e empilhar em **AUX.** Dentro de um laço de repetição:
       - Desempilhar um elemento da pilha **P** e empilhar na pilha **AUX**.
       - Verificar se o elemento desempilhado é igual **v.** Caso afirmativo, o valor **v** está presente na pilha P e, no final da função, o valor *true* deve ser retornado.
     - Devolver os elementos armazenados em **AUX** para **P**. Dentro de um laço de repetição:
       - Desempilhar um elemento da pilha **AUX** e empilhar na pilha **P**.
     - Retorna true ou false
   - Exemplo:

```
bool buscaValor(Pilha *p, int valor)
{
      //criar e inicializar a pilha AUX
      bool encontrou=false;
      while(! vaziaP(p))
      {
            V = desempilha(p);
            Empilha(&aux, v);
            If(v== valor)
            {
                  encontrou=true;
                  break;
            }
      }

      //implementar o laço de repetição para devolver os elementos
      //armazenados em AUX para P

      return encontrou;
}
```

7. Crie uma pilha que permita armazenar dados do tipo string. Para isso, deve-se modificar a estrutura da Pilha, “int *dados;” para “string *dados;”. Além disso, as demais funções que recebem o valor do tipo inteiro devem ser alteradas para o tipo string. Exemplos:

```
void inicializaP(Pilha *p, int tam)
{
    .
    .
    .
    p->dados = new string[tam];///aloca memória dinamicamente
}
bool empilhaP(Pilha *p, string valor)
string desempilhaP(Pilha *p)
string espiaP(Pilha *p)
bool buscaP(Pilha *p, string valor)
```

8. Faça um programa que leia uma string e verifica se o texto é um palíndromo, ou seja, se a string é escrita da mesma maneira de frente para trás e de trás para frente. Ignore espaços e pontos. Exemplos: Osso, Radar, Ovo, Arara

   Observações:

   - Modificar a estrutura da Pilha para armazenas dados do tipo char.
   - Use uma pilha para realizar a verificação do palíndromo, não é permitido o uso de vetores auxiliares.