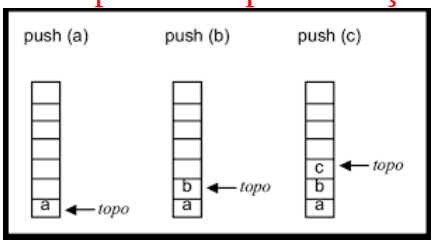
Alguns exercícios resolvidos: Lista 05

5. Escreva um programa para simular as operações de uma pilha (First in last out - FILO), com 10 posições. O primeiro elemento a ser empilhado é o último a ser retirado da pilha. Este tipo de estrutura é comumente utilizado para gerenciar chamadas de funções, por exemplo. O programa deve ter as operações de inserção e remoção. O processo de remoção não é físico. Uma remoção deve ser controlada por uma indicação lógica.

Exemplo ilustrativo de uma pilha – implementação por meio de um vetor



- Aplicações para gerenciar:
  - Funções recursivas em compiladores;
  - Recurso de Desfazer digitação em editores de texto;
  - Função voltar/retornar em navegadores, por exemplo.

Solução

```
#include<stdio.h>

#define n 10


int main()
{
  int pilha[n], op, topo=0;

  printf("\n\t - Programa para exemplificar um Tipo abstrato de dados (pilha) estático -");
  do{
    printf("\n\t_____________________________________________________________________");
    printf("\n\t\t\t\t 1. Push");
    printf("\n\t\t\t\t 2. Pop");
    printf("\n\t\t\t\t 3. Print");
    printf("\n\t\t\t\t 4. Sair");
    printf("\n\t_____________________________________________________________________");
    printf("\n\t\t\t\t Escolha uma opção: ");
    scanf("%d",&op);
    if (op<=0 || op>4)
      printf("\n \t\t\t \t Digite uma opção válida!");
    else
    {
      if (op==1)
      {
        if (topo<n)
        {

          printf("\n\t\t\t\t Digite um elemento para constar na pilha: ");
          scanf("%d",&pilha[topo]);
          topo++;
        }
        else
          printf("\n\t\t\t\tPilha cheia!");
      }
      else if (op==2)
      {
        if (topo>0)
          topo--;
        else
```

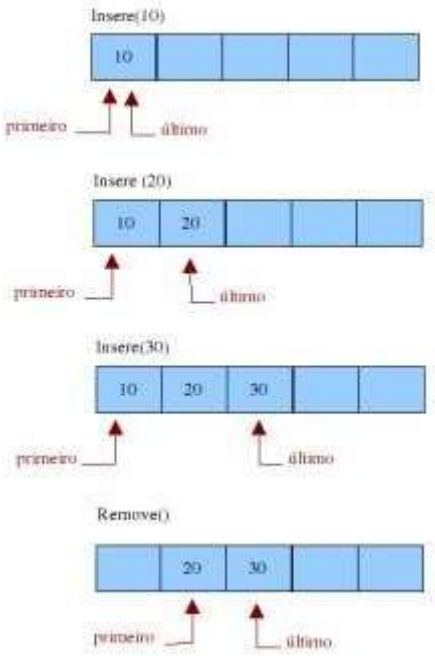
```
            printf("\n\t\t\t\tNão existe elemento para realizar POP!");

        }
        else if (op==3)
        {
          if (topo>0)
          {
            printf("\n\t\t\t\tElemento na pilha: %d", pilha[topo-1]);
          }
          else
            printf("\n\t\t\t\tPilha vazia!");
        }
      }
    }while(op!=4);


// system("pause");
  return 0;
}
```

6. Escreva um programa para simular as operações de uma fila (FIFO), ou seja, o primeiro elemento a entrar na fila é o primeiro a ser retirado da fila. O programa deve permitir uma fila com 10 posições. Uma lista FIFO é comumente utilizada para gerenciar processos em que a ordem de chegada é a que deve ser utilizada para o atendimento, por exemplo, como ocorre em uma fila de impressão. O programa deve ter as operações de inserção e remoção. O processo de remoção não é físico. Uma remoção deve ser controlada por uma indicação lógica.

Exemplo ilustrativo de uma Fila (FIFO) – Implementada por meio de um vetor



Exemplos de Aplicações:
- Gerenciar fila de impressão; -
- Gerenciar fila de processos - processador

Solução

```
#include <stdio.h>

#define n 10
```

```c
int main()
{
  int cont=0, ultimo=0, primeiro=0, op, fifo[n];

  do{

      printf ("\n\t\t\t ----- Programa FIFO -----");

      do{
            printf ("\n\t 1. Consultar primeiro da fila");
            printf ("\n\t 2. Incluir elemento na fila");
            printf ("\n\t 3. Atender primeiro da fila");
            printf ("\n\t 4. Sair\n");
            scanf("\t%d",&op);

      }while ( op < 1 || op>4);

       if (op == 1)
      {
           if ( cont == 0)
              printf ("\n\t\t\t -- Fila Vazia! --");
            else
              printf ("\n Primeiro elemento: %d",fifo[primeiro]);
      }
      else if( op == 2)
      {
           if (cont == n)
              printf ("\n\t\t\t -- Fila Cheia! --");
           else
           {
              printf ("\n Codigo:");
              scanf("%d",&fifo[ultimo]);
              ultimo++;
              cont++;
           }

       }
       else if (op ==3)
       {
           if(cont==0)
              printf ("\n\t\t\t -- Fila Vazia! --");
           else
           {
              primeiro++;
              cont--;
           }
       }
       printf("\n");
      // system ("pause");

   }while (op !=4);
   //system ("pause");
   return 0;
}
```

12. Dado um vetor VIN de 10 elementos inteiros, criar um vetor VAI de 30 elementos, sendo que VAI[0], VAI[1] e VAI[2] recebem o valor de VIN[0] e assim por diante. Escrever um programa que leia VIN e imprima VAI.

Solução

```
#include <stdio.h>

int main ()
{
  int VIN[10], VAI[30];

  for (int i=0; i<10; i++)
  {
    printf ("\nDigite o valor de VIN para i igual a %d\n", i);
    scanf ("%d", &VIN[i]);
    VAI[3*i]=VIN[i];
    VAI[3*i+1]=VIN[i];
    VAI[3*i+2]=VIN[i];
  }

  for (int i=0; i<30; i++)
  {
    printf ("\nVAI %d tem valor de %d", i, VAI[i]);
  }

// system ("PAUSE");
  return 0;
}
```

14. Faça um programa que leia um vetor X capaz de armazenar cinco números inteiros. Em seguida, o programa deve dividir todos os elementos contidos em X pelo maior valor do vetor. Mostre o vetor antes e após os cálculos.

Solução

```
#include <stdio.h>

//Definir previamente a dimensão do vetor
#define max 5
int main()
{

  int n[max],mn=0;
  for(int i=0; i<max; i++)
  {
    printf("\nDigite o %do. valor inteiro:\n",i+1);
    scanf("%d",&n[i]);

    if(i==0) //inicializar a variável para o maior valor do vetor
      mn=n[i];
    else
    {
      if(n[i]>mn)
        mn=n[i];
    }
  }
```

```
  printf("\n");
  printf("\nMaior número %d:", mn);

  for(int i=0; i<max; i++){
    printf("\n%d / %d = %.2f\n",n[i],mn,(float)n[i]/mn);
  }
  //system("pause");
  return 0;
}
```

15. Elaborar um programa para calcular o reajuste de salário de uma empresa com 10 funcionários. O usuário deve responder se deseja continuar ou não a execução do algoritmo. Considere que o funcionário deverá receber um reajuste de 15%, caso seu salário seja menor que R$ 500,00. Se o salário for maior ou igual a R$ 500,00, mas menor ou igual a R$ 1000,00, o reajuste será de 10%. Se salário for maior que R$ 1000,00, o reajuste deverá ser de 5%. Apresentar uma lista no seguinte formato:

Lista de Salários dos Funcionários da Empresa AB			
Código	**Nome**	**Salário Base**	**Salário Reajustado**
	Total:		

```
//Diretivas de Pré-processamento (Obrigatórias)
#include <stdio.h>
#include <stdlib.h>
//Ajustar n para 10 após os testes
#define n 3
//Obrigatório. Função principal: indica o início da execução do programa
int main ()
{
  //Declaração de Variáveis Locais.
  int cod_func[n];
  float sal_func[n], sal_reajuste[n], total_sal_base=0, total_sal_reaj=0;
  char nome_func[n][40];

  printf("\n\n\t\t - Programa Reajuste de Salário -");
  for (int i=0; i<n; i++)
  {
    //Obrigatório. Comandos para resolução do problema
    printf("\n\n\tDigite o código do funcionário: ");
    scanf("%d",&cod_func[i]);
    printf("\n\tDigite o salário do funcionário: ");
    scanf("%f",&sal_func[i]);
    printf("\n\tDigite o nome do funcionário: ");
    scanf(" %[^\n]s",nome_func[i]);
    //setbuf(stdin,NULL);
    //Acumula salário base
    total_sal_base+=sal_func[i];
```

```
    if (sal_func[i]<500)
      sal_reajuste[i]=sal_func[i]*1.15;
    else if (sal_func[i]>=500 && sal_func[i]<=1000)
      sal_reajuste[i]=sal_func[i]*1.10;
    else
      sal_reajuste[i]=sal_func[i]*1.05;
    //Acumula salário reajustado
    total_sal_reaj+=sal_reajuste[i];

  }
  printf("\n\n\t_____________________________________________________________
_____________");
  printf("\n\t\t\tLista de Salários dos Funcionários da Empresa AB");
  printf("\n\t_____________________________________________________________
__________");
  printf("\n\t\tCodigo\tNome\t\t\t\tSal. Base\tSalario Reajustado");
  printf("\n\t_____________________________________________________________
__________");

  for (int i=0; i<n; i++)
  {
    printf("\n\t\t%d\t%s\t\t\t\t%.2f\t\t\t%.2f", cod_func[i],nome_func[i],
sal_func[i],sal_reajuste[i]) ;
  }

  printf("\n\n\t_____________________________________________________________
_____________");
  printf("\n\n\t\t\t\t\t\tTotal:|%.2f\t\t\t|%.2f",total_sal_base,total_sal_reaj);
  printf("\n\n\t_____________________________________________________________
_____________\n");
    //Opcional. Comando para interromper momentaneamente o programa
   //system("PAUSE");
    //Retorno ao SO o status do programa
    return 0;
}//Indica o final do programa.
```

20. Seja A e B dois vetores contendo 10 elementos inteiros. Fazer um programa para:

a) ler A e B.
b) Calcular a soma dos elementos de A.
c) Calcular a soma dos elementos de B.
d) Obter o vetor C, que é a soma dos vetores A e B.
e) Obter o vetor D, subtraindo B de A.

```c
#include <stdio.h>

#define max 10

int main()
{
  int vetor_A[max], vetor_B[max], vetor_C[max], vetor_D[max], soma_A=0, soma_B=0;

  for(int i=0; i<max; i++)
  {
    printf("Digite um valor para a posição %d, vetor A: \n", i);
    scanf("%d", &vetor_A[i]);

    soma_A=soma_A+vetor_A[i];

    printf("Digite um valor para a posição %d, vetor B: \n", i);
    scanf("%d", &vetor_B[i]);

    soma_B=soma_B+vetor_B[i];

    vetor_C[i]=vetor_A[i]+vetor_B[i];
    vetor_D[i]=vetor_B[i]-vetor_A[i];
  }

  printf("Soma dos valores do vetor A: %d\n", soma_A);
  printf("Soma dos valores do vetor B: %d\n", soma_B);

  printf("--------------------Vetor C--------------------\n");

  for(int i=0; i<max; i++)
    printf("Vetor C[%d] -> soma dos vetores A e B: %d\n",i,vetor_C[i]);

  printf("--------------------Vetor D--------------------\n");
  for(int i=0; i<max; i++)
    printf("Vetor D[%d] -> subtracao dos vetores B e A: %d\n",i,vetor_D[i]);

  //system("PAUSE");
  return 0;
}
```