,

Universidade do Extremo Sul Catarinense
Unidade Acadêmica de Ciências, Engenharias e Tecnologias
Curso de Ciência da Computação
Disciplina: 7532 - Estrutura de Dados I
Professora: Sergio Coral

Avaliação 1 - Listas Lineares (Peso 10)

Nome do Aluno:_____GAbarito_____**Data: 16/04/2019**
Observações

- Não esqueça de colocar o seu **nome**
- A avaliação deve ser entregue totalmente escrita a **caneta**

Questões:

1) Considerando que uma lista linear pode ser representa em C conforme abaixo.

```
#define TAM 10
struct lista {
        int ultimo;
        int elementos[TAM];
};

int main(){
    lista lis;
    criaLista(&lis);
   insereInicio(&lis,10);
   consultaLista(&lis);
   ordenaAsc(&lis);
}
```

a) (2,00) Implemente a função insereInicio para adicionar um novo elemento no início da lista.

```
void insereInicio(lista *lis, int valorIns){
    if(lis->ultimo>=TAM-1){
      cout << "Nao é possivel inserir no inicio pois a lista esta cheia." << endl;
      return;
    }

    //incrementa 1 à variável que mantém o valor do último elemento
    lis->ultimo++;

    //desloca todos os elementos a partir do início em uma casa no vetor
    for(int i = lis->ultimo; i > 0; i--){
      lis->elementos[i] = lis->elementos[i - 1];
    }

    //insere o elmento passado por parâmetro no início da lista
    lis->elementos[0] = valorIns;
}
```

b) (2,00) Implemente a função consultaLista para apresentar todos os dados mantidos na lista.

```
void consultaLista(lista *lis){
   if(listaVazia(lis->ultimo == -1)){
      cout << "Nao é possivel consultar pois a lista esta vazia." << endl;
      return;
   }

   //percorre todos os elementos da lista a partir do início apresentando os valores inseridos
   for(int i = 0; i <= lis->ultimo; i++){
      cout << "Valor [" << i << "]: " << lis->elementos[i] << endl;
   }
}
```

c) (2,00) Implemente a função ordenaAsc para ordenar os dados da lista em ordem crescente.

```
void ordenaAsc(lista *lis){
    for(int i = 0; i <= lis->ultimo; i++){
        //percorre a lista até o penúltimo elemento
        for(int j = 0; j < lis->ultimo; j++){
            int elAtual = lis->elementos[j];
            int elProx  = lis->elementos[j + 1];
            if(elAtual > elProx){

                //altera o elemento atual, informando o próximo
                lis->elementos[j] = elProx;

                //altera o próximo elemento, informando o atual
                lis->elementos[j + 1] = elAtual;
            }
        }
    }
}
```

---

2) (2,0) Seja o seguinte vetor, ordenado de forma ascendente. Caso se utilize um algoritmo de busca binária, quantas iterações serão necessárias para que cada um dos valores seja encontrado?

Valor	30	35	40	45	50	55	60	65	70	80	83
Índice	0	1	2	3	4	5	6	7	8	9	10

4.1) Valor 70	4.2) Valor 30	4.3) Valor 45	4.4) Valor 80
a) (X) 2 b) ( ) 3 c) ( ) 4 d) ( ) 8 e) ( ) 9	a) ( ) 2 b) (X) 3 c) ( ) 4 d) ( ) 8 e) ( ) 9	a) ( ) 2 b) (X) 3 c) ( ) 4 d) ( ) 8 e) ( ) 9	a) ( ) 2 b) (X) 3 c) ( ) 4 d) ( ) 8 e) ( ) 9

3) (2,0) Considerando a implementação de operações para o controle de um TAD do tipo lista, analise o código das funções abaixo e defina de forma objetiva o que ela faz.

void misterio_A (**lista** *lis, **int** pos){
if(listaVazia(lis) || pos > lis->ult){
return;
}
 	
for(int i = pos; i < lis->ult; i++){
lis->valor[i] = lis->valor[i + 1];
}
lis->ult--;
}
	void misterio_B (**lista** *lis, int valorIns){
	if(listaCheia(lis)){
	return;
	}
	
	lis->ult++;
	for(int i = lis->ult; i > 0; i--){
	lis->valor[i] = lis->valor[i - 1];
	}
	
	lis->valor[0] = valorIns;
	}

R: removePosicao= Remove um elemento na posição estabelecida.	**R: insereInicio = Inseri o valor no início da lista**