Universidade Federal do ABC	**Avaliação**: Prova 02 (40 pontos)
Disciplina: MCTA028 - Programação Estruturada	**Turma**: Noturno – A1 e A2
Professor: Jesús P. Mena-Chalco	**Data**: 06/12/2017

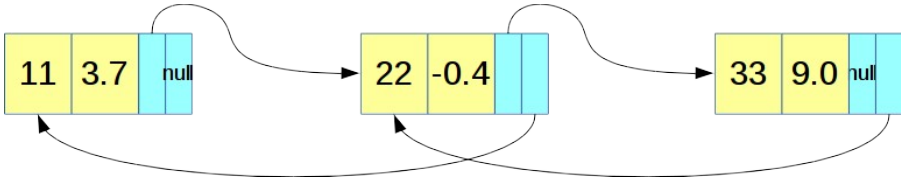
Nome completo:		RA:

Instruções:

- Responda cada questão no espaço reservado.
- Preencha seu nome e RA a caneta; o resto pode ser a lápis.

1. Defina uma estrutura, na linguagem C, que seja a base para implementar uma **lista duplamente ligada** que armazene um número real **e** um número inteiro.
   Desenhe uma lista duplamente ligada contendo 3 elementos. [3 pontos]

```
// Uma possível solução:
struct celula {
    int    conteudoInteiro
    long   conteudoReal;
    struct celula *seguinte;
    struct celula *anterior;
};
```



2. Considere o programa abaixo que permite calcular o valor de Pi usando os primeiros `t` termos da aproximação por série de Gregory:

$$\frac{\pi}{4} = 1 - \frac{1}{3} + \frac{1}{5} - \frac{1}{7} + \frac{1}{9} - \dots$$

```
double pi_gregory (int t) {
    int i, sinal=1;
    double soma=0;

    for (i=1; i<=t; i++) {
        soma  += 1.0/(2*i-1)*sinal;
        sinal *= -1;
    }
    return 4.0*soma;
}
```

[3 pontos]

Dado um inteiro `t`, qual o **número total de multiplicações** (*) necessárias para calcular a aproximação de Pi?

```
3*t + 1
```

3. Considere o programa abaixo que permite ordenar, na forma crescente, um vetor de n números inteiros.

```
void BubbleSort2 (int v[], int n) {
    int i, aux, hasChanged;
    do {
        hasChanged = 0;
        for (i=0; i<n-1; i++) {
            if (v[i] > v[i+1]) {
                aux    = v[i];
                v[i]   = v[i+1];
                v[i+1] = aux;
                hasChanged = 1;
            }
        }
    } while(hasChanged==1);
}
```

[4 pontos]

Indique o **número de trocas** (swaps) entre elementos do vetor v

- No pior caso:
  n(n-1)/2 trocas
  Quando o vetor está ordenado na sua forma decrescente.

- No melhor caso:
  0 trocas
  Quando o vetor está ordenado na sua forma crescente.

4. Considere o programa abaixo que permite ordenar, na forma crescente, um vetor de n números inteiros.

```
void Cocktailsort (int v[], int n) {
    int i, t, trocou=1;
    int inicio=0, fim=n-1;

    while (trocou==1) {
        trocou = 0;
        for (i=inicio; i<fim; i++) {
            if (v[i] > v[i+1]) {
                t      = v[i];
                v[i]   = v[i+1];
                v[i+1] = t;
                trocou = 1;
            }
        }
        fim--;

        if (trocou==0)
            break;

        trocou = 0;
        for (i=fim-1; i>=inicio; i--) {
            if (v[i] > v[i+1]) {
                t      = v[i];
                v[i]   = v[i+1];
                v[i+1] = t;
                trocou = 1;
            }
        }
        inicio++;
    }
}
```

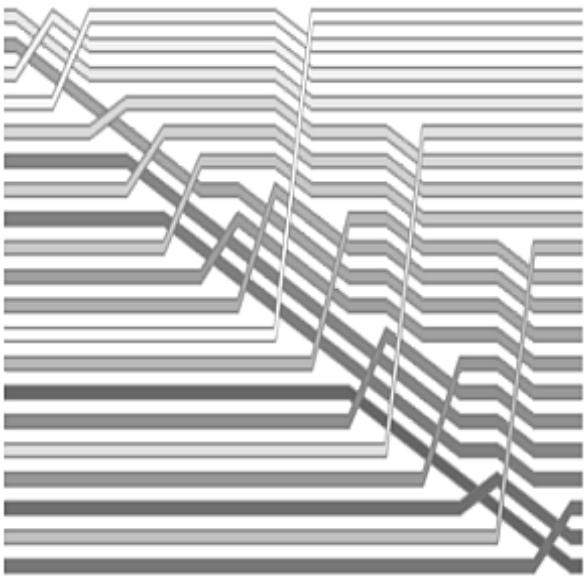
[4 pontos]

Indique o **número exato** de comparações entre elementos do vetor

- No pior caso:
  n(n-1)/2 comparações
  Quando o vetor está ordenado na sua forma decrescente.

- No melhor caso:
  n-1 comparações
  Quando o vetor está ordenado na sua forma crescente.

5. Considere o diagrama abaixo que representa de forma visual as trocas de elementos em um vetor desordenado para torná-lo ordenado na sua forma crescente.



[6 pontos]

- Qual é o **nome** do algoritmo de ordenação?
  Insertion Sort

- Para esse algoritmo: Qual seria a sequência de elementos no vetor que **maximizaria** o número de trocas?
  A sequência de elementos no vetor deve estar ordenada na forma **decrescente.**

- Para esse algoritmo: Qual seria a sequência de elementos no vetor que **minimizaria** o número de trocas?
  A sequência de elementos no vetor deve estar ordenada na forma **crescente.**

6. Considere o programa abaixo:

```c
#include <stdio.h>
#include <stdlib.h>

int f1(int *p, int *r) {
  if (p==r)
      return 0;
  else {
      r--;
      return *r + f1(p, r);
  }
}

int main() {
  int i, j, n;
  scanf("%d", &n);

  int *p = (int *) malloc(n*n*sizeof(int));

  for (i=0; i<n; i++)
      for (j=0; j<n; j++)
          scanf("%d", p+i*n+j);

  printf("\n%d\n", f1(p, p+n*n) );

  return 0;
}
```

[6 pontos]

Explique sucintamente o que o programa faz:

O programa imprime a **somatória de todos os elementos de uma matriz** quadrada de ordem n, dada como entrada.

7. Considere o programa abaixo:

```c
#include <stdio.h>
#include <stdlib.h>

struct celula {
  long   conteudo;
  struct celula *seguinte;
};

int main() {
  struct celula *x, *y, *t;

  x = malloc(sizeof(struct celula));
  y = malloc(sizeof(struct celula));

  x->seguinte = y;
  x->conteudo = 1;
  y->seguinte = x;
  y->conteudo = 1;

  for (t=x; t->conteudo < 50; t=t->seguinte) {
      t->conteudo = x->conteudo + y->conteudo;
      printf("%ld\n", t->conteudo);
  }
  return 0;
}
```

[6 pontos]

Explique sucintamenteo que o programa faz:

O programa, usando uma lista encadeada circular, imprime a **sequênca de Fibonacci** com inicio em 2 até 89:

```
2
3
5
8
13
21
34
55
89
```

8. Crie um algoritmo que permita realizar a busca de um elemento chave usando o algoritmo de busca binária em um vetor v[0,...n-1] ordenado na forma crescente. Se o elemento chave estiver deve-se devolver o índice no vetor v. Caso contrário deve-se devolver o índice do elemento mais próximo ao elemento procurado. [4 pontos]

```
int buscaBinariaRec2 (int chave, int A[], int inf, int sup) {
    int meio = (inf+sup)/2;

    if (inf>sup) {
        if ( abs(A[meio]-chave)<abs(A[meio+1]-chave) )
            return meio;
        else
            return meio+1;
    }

    if (chave==A[meio])
        return meio;

    if (chave<A[meio])
        return buscaBinariaRec2(chave, A, inf, meio-1);
    else
        return buscaBinariaRec2(chave, A, meio+1, sup);
}
```

9. O que há de errado com a seguinte função recursiva? [4 pontos]

```
int XX(int n) {
    if (n==0)
        return 0;
    else
        return XX(n/3 + 1) + n;
}
```

O programa, não tem os casos base bem definidos.

Para valores positivos e diferentes de zero o programa não finaliza devido aos chamados recursivos sem fim. Dado o limite da pilha do computador a execução do programa finalizará por estouro de pilha (stack overflow).

Em particular, para n=1, é realizado um outro chamado passando como parâmetro o mesmo valor.

Por exemplo:

```
  XX(1) = XX(1/3+1) + 1
        = XX(1) + 1
```