INF 1620 – P2 - 01/11/03	Questão 1
Nome:	
Matrícula:	Turma

Considere a implementação de uma lista encadeada para armazenar as notas dos alunos de uma turma dada pelo tipo abaixo:

```
struct lista {
   char nome[81];
   int mat;
   float p1, p2, p3;
   struct lista* prox;
};
typedef struct lista Lista;
```

Implemente uma função que, dados uma lista encadeada e o número de matrícula de um aluno, tenha como valor de retorno a média obtida pelo aluno nas três provas (*m = (p1+p2+p3)/3*). Se não for encontrado nenhum aluno com o número de matrícula fornecido, a função deve retornar o valor -1.0. Essa função deve obedecer o seguinte protótipo:

```
float media (Lista* l, int matric);
```

```
float media (Lista* l, int matric) {
   Lista* p;
   for (p = l; p!=NULL; p = p->prox)
      if (p->mat == matric)
         return (p->p1 + p->p2 + p->p3)/3;
   return -1.0;
}
```

Não separe as folhas deste caderno. Todas as folhas devem ter seu nome. Responda cada questão na folha correspondente. Use o verso se necessário.

INF 1620 – P2 - 01/11/03	Questão 2
Nome:	
Matrícula:	Turma

Considerando uma estrutura de pilha para armazenar valores inteiros, implemente a função para inserir um novo elemento na pilha dada pelo protótipo:

```
void push (Pilha* p, int v);
```

a) Utilizando a implementação com uso de vetor com o tipo abaixo:

```
#define MAX 100
struct pilha {
   int n;
   int info[MAX];
};
typedef struct pilha Pilha;
```

b) Utilizando a implementação com uso de lista encadeada com os tipos abaixo:

```
struct lista {
   int info;
   struct lista* prox;
};
typedef struct lista Lista;
struct pilha {
   Lista* topo;
};
typedef struct pilha Pilha;
```

Obs: Assuma que em ambos os casos sempre será possível inserir o elemento na pilha.

```
a)
void push (Pilha* p, int v) {
   p->info[p->n] = v; /*assumindo que n indica a*/
   p->n++;            /*primeira posição livre do vetor*/
}

b)
void push (Pilha* p, int v) {
   Lista* novo = (Lista*) malloc(sizeof(Lista));
   novo->info = v;
   novo->prox = p->topo;
   p->topo = novo;
}
```

Não separe as folhas deste caderno. Todas as folhas devem ter seu nome. Responda cada questão na folha correspondente. Use o verso se necessário.

INF 1620 – P2 - 01/11/03	Questão 3
Nome:	
Matrícula:	Turma

Considere a existência de um tipo abstrato `Fila` de números reais, cuja interface é definida pelo arquivo `fila.h` com o seguinte conteúdo:

```
typedef struct fila Fila;

Fila* cria (void);                /* cria uma fila vazia */
void insere (Fila* f, float v);   /* insere um elemento */
float retira (Fila * f);          /* retira um elemento */
int vazia (Fila * f);             /* retorna se está vazia */
void libera (Fila * f);           /* libera a estrutura */
```

Sem conhecer a representação interna deste tipo abstrato e usando apenas as funções declaradas no arquivo `fila.h`, implemente uma função que, dados uma fila e um valor real `x`, crie uma nova fila com apenas os elementos maiores que `x`. Por exemplo, se a fila original tiver os elementos `4.8,3.2,9.8,3.3` e for fornecido o valor `x=4.3`, a fila nova criada deve conter apenas os elementos `4.8,9.8`. Observe que a ordem dos elementos deve ser preservada. Ao final da função, a fila original deve estar vazia e a função deve retornar a nova fila criada. O protótipo da função deve ser:

```
Fila* filtra (Fila* f, float x);
```

```
Fila* filtra (Fila* f, float x) {
   Fila* f2 = cria();
   while (!vazia(f)) {
      float v = retira(f);
      if (v > x)
         insere(f2,v);
   }
   return f2;
}
```

Não separe as folhas deste caderno. Todas as folhas devem ter seu nome. Responda cada questão na folha correspondente. Use o verso se necessário.

INF 1620 – P2 - 01/11/03	Questão 4
Nome:	
Matrícula:	Turma

Considerando uma estrutura de árvore binária definida pelo tipo abaixo:

```
struct arv {
   char info[81];
   struct arv* esq
   struct arv* dir;
};
typedef struct arv Arv;
```

Escreva uma função que verifique se uma árvore é cheia. Uma árvore é dita cheia se todos os nós que não são folhas têm os dois filhos, isto é, não pode existir nó com apenas um filho. A função deve retornar 1 no caso da árvore ser cheia ou 0 no caso de não ser. O protótipo da função deve ser:

```
int cheia (Arv* a);
```

Obs: No caso da árvore ser vazia, a função também deve retornar 1.

```
int cheia (Arv* a) {
   if (a == NULL)
      return 1;
   else if ((a->esq == NULL && a->dir != NULL) ||
            (a->esq != NULL && a->dir == NULL))
      return 0;
   else
      return cheia(a->esq) && cheia (a->dir);
}
```

Não separe as folhas deste caderno. Todas as folhas devem ter seu nome. Responda cada questão na folha correspondente. Use o verso se necessário.

INF 1620 – P2 - 01/11/03	Questão 5
Nome:	
Matrícula:	Turma

Considerando a estrutura de árvore genérica que armazena valores inteiros definida pelo tipo abaixo:

```
struct arvgen {
   int info;
   struct arvgen *prim;  /* ponteiro p/ primeiro filho */
   struct arvgen *prox;  /* ponteiro p/ irmão */
};
typedef struct arvgen ArvGen;
```

Escreva uma função que retorne o maior valor inteiro armazenado na árvore. Considere que a árvore dada não é vazia. A função deve seguir o protótipo:

```
int maximo (ArvGen* a);
```

```
int maximo (ArvGen* a) {
   int max = a->info;
   ArvGen* p;
   for (p = a->prim; p!=NULL; p = p->prox) {
      int v = maximo(p);
      if (v > max)
         max = v;
   }
   return max;
}
```

Não separe as folhas deste caderno. Todas as folhas devem ter seu nome. Responda cada questão na folha correspondente. Use o verso se necessário.

RASCUNHO

Respostas nesta folha não serão consideradas.

Protótipos de funções que podem ser úteis:

stdio.h:

```
int scanf (char* formato, ...);
int printf (char* formato, ...);
```

math.h:

```
double sqrt (double x);
double pow (double x, double exp);
```

string.h:

```
int strlen (char* s);
int strcmp (char* s, char *t);
char* strcpy (char* destino, char* fonte);
char* strcat (char* destino, char* fonte);
```

sdtlib.h:

```
void* malloc (int nbytes);
void free (void* p);
```

Não separe as folhas deste caderno.