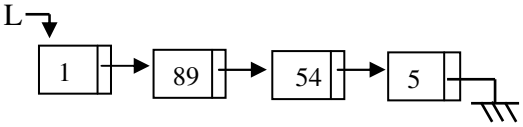
INF 1620 – P3 - 21/06/08	Questão 1
Nome:	
Matrícula:	Turma

Dada uma lista encadeada de números inteiros cujo tipo que representa um nó da lista é dado por:

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

Implemente uma função que receba uma lista encadeada do tipo `Lista` e retorne um vetor de números inteiros, alocado dinamicamente, contendo as informações presentes na mesma ordem que na lista, e o número de elementos neste vetor.

Assim, se for recebida a lista



será retornado o vetor `{1, 89, 54, 5}` e o número de elementos igual a 4. Se a lista for vazia, o vetor retornado deverá ser NULO (`NULL`) e o número de elementos igual a 0.

O protótipo da função é dado por:

```
int* Copia_Lista(Lista* L, int* n);
```

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

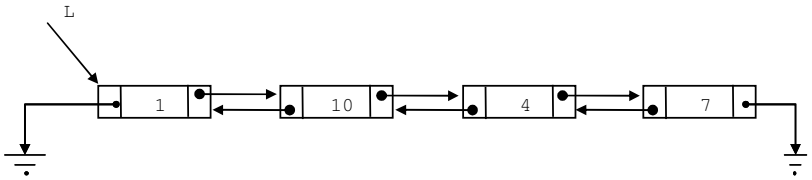
INF 1620 – P3 - 21/06/08	Questão 2
Nome:	
Matrícula:	Turma

Considere estruturas de listas duplamente encadeadas que armazenam valores inteiros. O tipo que representa um nó da lista é dado por:

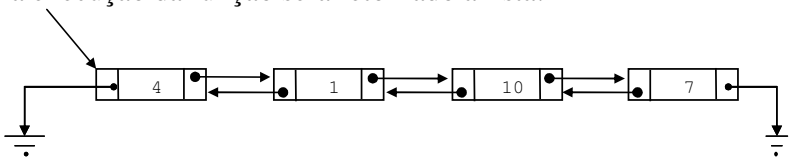
```
struct lista_d {
       int info;
       struct lista_d* prox;
       struct lista_d* ant;

};
typedef struct lista_d Lista_d;
```

Escreva uma função que recebe como parâmetro o ponteiro para o primeiro nó da lista duplamente encadeada `L` e um número `n`, retornando uma nova lista resultante da movimentação do enésimo elemento da lista para a primeira posição da lista.

Seja a lista



e `n` igual a 3, após a execução da função será retornado a lista:



O protótipo da função é dado por:

```
Lista_d* Altera_Pos(Lista_d* L, int n);
```

Observação: Se `n` for maior que o número de nós da lista ou se `n` for menor ou igual a `0`, a lista deve ser retornada inalterada.

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 – P3 - 21/06/08	Questão 3
Nome:	
Matrícula:	Turma

Uma árvore binária de busca é definida com a propriedade de que em qualquer nó `n`, todos os elementos da sub-árvore esquerda de `n` têm valores menores que o de `n`, e todos os elementos da sub-árvore direita têm valores maiores ou iguais que o de `n`.

Dado o tipo correspondente:

```
typedef struct abb Abb;
struct no {
       int val;
       Abb* esq;
       Abb* dir;
};
```

Escreva uma função com protótipo:

```
int Conta_Maiores(Abb* a, int x);
```

que retorne o números de nós da árvore cujo o valor é maior que x. No caso da árvore ser vazia deverá ser retornado o valor 0.

Observação: A implementação deve levar em consideração a ordenação dos dados na árvore binária de busca

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 – P3 - 21/06/08	Questão 4
Nome:	
Matrícula:	Turma

Suponha a seguinte declaração, relativa a um tipo "árvore genérica", ou seja, uma árvore cujos nós têm um número não limitado de filhos, que armazena nomes de arquivos ou diretórios.

```
struct arvv {
 char nome_recurso[101]; /* nome do arquivo ou diretorio */
 struct arvv* prim; /* ponteiro para o primeiro filho do diretorio */
 struct arvv* prox; /* ponteiro para o proximo irmao do arquivo/diretorio */
};

typedef struct arvv ArvVar;
```

Escreva uma função que recebe um ponteiro para uma árvore de diretórios, o nome de um diretório e o nome do outro recurso (arquivo ou diretório), e retorna 1 se, na árvore de diretórios, o recurso está diretamente contido no diretório (isto é, se o recurso é um nó filho do diretório), ou 0 caso contrário. A função deve ter o seguinte protótipo:

```
int Verifica_Recurso (ArvVar *A, char *Diretorio, char * Recurso);
```

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, ...);
FILE* fopen (char* nome, char* modo);
int fclose (FILE* fp);
int fscanf (FILE* fp, char* formato, ...);
int fprintf (FILE* fp, char* formato, ...);
char* fgets(char* str, int size, FILE* fp));
int sscanf(char* str, char* formato, ...);
```

math.h:

```
double sqrt (double x);
double pow (double x, double exp);
double cos (double radianos);
double sin (double radianos);
```

string.h:

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

stdlib.h:

```
void* malloc (int nbytes);
void free (void* p);
void qsort (void *vet, int n, int tam, int (*comp) (const void*, const void*));
```

Não separe as folhas deste caderno.