INF 1620 – P1 - 10/04/01	Questão 1

Escreva uma função `troca` para trocar dois caracteres. Por exemplo, se tivermos `c1='a'` e `c2='b'`, antes da chamada de `troca`, teremos `c1='b'` e `c2='a'`, após a chamada de `troca`.
Justifique sua resposta, explicando o funcionamento de sua função.

```
void troca(char *c, char *d) {
   char t=*c;
   *c=*d;
   *d=t;
}
```

Note que a variável `t` é do tipo `char`, e não do tipo `char *`. Esta é uma variável temporária, que guarda o caracter apontado por `c` enquanto a troca dos valores está sendo feita.

Para ver como a função pode ser chamada, considere o exemplo:

```
char c1='a', c2='b';
printf("c1=%c; c2=%c;\n",c1, c2);
                                  /* escreve c1=a; c2=b; */
troca(&c1,&c2);
printf("c1=%c; c2=%c;\n",c1, c2);
                                  /* escreve c1=b; c2=a; */
```

Na chamada de troca, são passados os endereços de `c1` e de `c2`.

INF 1620 – P1 - 10/04/01	Questão 2

Escreva uma função que passe uma cadeia de caracteres para maiúsculas, isto é, que substitua todos as letras minúsculas pelas letras maiúsculas correspondentes, deixando os demais caracteres inalterados.

Uma solução possível é:

```
#include <string.h>
void converte (char * s) {
   char c;
   int i, l=strlen(s);
   for (i=0; i<=l; i++) {
      c=s[i];
      if ((c>='a') && (c<='z'))  /* c é minúscula */
         s[i]=c-'a'+'A';         /* converte */
   }
}
```

Nota: As bibliotecas padrão da linguagem C oferecem funções que podem facilitar a conversão. RTFM.

INF 1620 – P1 - 10/04/01	Questão 3

Escreva um programa em C com a seguinte finalidade: o programa recebe como entrada (lê do teclado) uma cadeia de caracteres `s`, e escreve (em uma linha) os 10 primeiros caracteres de `s`. Se o comprimento da cadeia `s` for menor do que 10, deve ser escrita uma linha com a cadeia `"***ERRO***"`.

```
#include <stdio.h>
#include <string.h>

int main(void) {
   char entrada[120]; /* chega? */
   int i, l;
   printf("cadeia: ");
   scanf("%[^\n]",entrada);
          /* formato qualquer coisa menos \n */
   l=strlen(entrada);
   if (l>=10) {
      for (i=0; i<10; i++)
         printf("%c",entrada[i]);
      printf("\n");
   } else
      printf("***ERRO***\n");
   return 0;
}
```

Uma outra solução aproveita o fato de que a cadeia `entrada` não será mais utilizada:

```
   if (l>=10) {
      s[10]='\0';
      printf("%s\n",entrada);
   } else
      printf("***ERRO***\n");
```

Note que a inserção de um caracter nulo efetivamente trunca a cadeia naquele ponto.

INF 1620 – P1 - 10/04/01	Questão 4

Suponha as seguintes declarações

```
#define MAX 100
struct s_pilha{
   int num;
   char vet[MAX};
};
typedef struct s_pilha *PILHA;
char topo(PILHA p);
void remove(PILHA p);
```

Escreva as funções `topo` *e* `remove`*, de maneira que:*

(a) O resultado da função `topo` *é o valor do elemento do topo da pilha.*

(b) A função `remove` *retira da pilha o elemento do topo.*

Observações:

- *O efeito combinado destas duas funções é o mesmo da função* `pop`*.*
- *Suponha que estas duas funções não serão chamadas se a pilha estiver vazia.*
- *Além disso, note que a pilha não deve ser alterada pela função* `topo`*.*
- *Justifique suas respostas, explicando o funcionamento das duas funções.*

A função `topo`, como especificada, apenas consulta o elemento do topo, que não é removido; esta é a finalidade da função `remove`, que apenas "decapita" a pilha, ignorando o elemento removido.

Podemos fazer:

```
char topo(PILHA p) {
   return p->vet[p->num-1];
}
```

Note que esta função não altera o valor de `p->num`, como aconteceria se usássemos `p->num--`.

```
void remove(PILHA p) {
   p->num--
}
```

INF 1620 – P1 - 10/04/01	Questão 5

Escreva uma função com protótipo

```
char *fecha(char *s);
```

que recebe como entrada uma cadeia de caracteres `s`*, e tem como resultado (retorna) uma cópia da cadeia entre colchetes. Por exemplo, se a cadeia* `s` *tem como valor "*`obaoba!`*", o resultado da função deve ser "*`[obaoba!]`*".*

A função fecha usa uma cadeia auxiliar `t`, que será o resultado da função.

```
#include <string.h>

char *fecha(char *s) {
   char *t= (char *) malloc(strlen(s)+3);
                     /* os 3 caracteres são: [ ] \0 */
   strcpy(t,"[");    /* copia [ para t */
   strcat(t,s);      /* acrescenta s a t */
   strcat(t,"]");    /* acrescenta ] a t */
   return t;         /* t é o resultado */
}
```

Uma alternativa mais compacta, e menos legível é

```
char *fecha(char *s) {
   char *t= (char *) malloc(strlen(s)+3);
   return strcpy(strcat(strcat(t,"["),s),"]");
}
```

(corrigido em 11/04/01)