Algoritmos e Programação de Computadores

2ª Prova

Questão	Valor	Nota
1	2,5	
2	2,5	
3	2,5	
4	2,5	
Total	10,0	

Nome:

RGA:

Instruções: Não é permitida consulta a qualquer material. *Somente serão consideradas respostas nos espaços marcados.* Use os versos das folhas como rascunho.

1. Considere o código a seguir e responda a pergunta abaixo. Observe que os valores da variáveis j e k na função *main* correspondem aos dois últimos dígitos de **seu** RGA, respectivamente. Assim, se o seu RGA é 201121902014, então $j = 1$ e $k = 4$.

```
#include <stdio.h>

void E(int *ri) {
  *ri = *ri + 3;
}

void F(int vj, int *rk) {
  int i = *rk;
  E(&i);
  printf ("%d %d %d\n",i, *rk, vj);
}

int main() {
  long rga;
  int j, k;
  scanf("%ld", &rga);
  j = (rga/10)%10;
  k = rga%10;
  F(j, &k);
  printf("%d %d %d\n", j, k, k+3);

  return 0;
}
```

a) Mostre a saída da execução deste programa para o **seu RGA**.

2. Escreva uma função chamada `união` que recebe dois conjuntos de mesmo tamanho `n`, armazenados nos vetores `A` e `B`. Esta função deve calcular a união entre `A` e `B` e armazenar o resultado no vetor `C`. Desta maneira, `C` deverá conter os elementos que estiverem em `A` e em `B`, desconsiderando as repetições. A função deverá devolver o tamanho do vetor `C`.

```
int uniao(int A[], int B[], int C[], int n) {

}
```

3. Escreva uma função que recebe um vetor de números inteiros *v* e o seu tamanho *n* e verifica se *v* está ordenado. Se *v* estiver ordenado, sua função deve devolver 1 e, 0, caso contrário.

```
int esta_ordenado(int v[], int n) {

}
```

4. Serviço de mensagens curtas (= *Short Message Service*, SMS) é um serviço disponível em telefones celulares que permite o envio de pequenas mensagens entre diferentes dispositivos. É notório que, para escrever estas mensagens em aparelhos que não possuem teclados alfabéticos, é necessário realizar algumas combinações com as teclas numéricas. Neste caso, deve-se observar que cada número é associado à um conjunto de letras (veja **Figura 1**).

Figura 1 - Leiaute de um teclado não alfabético.

Escreva uma função que recebe uma cadeia de caracteres contendo uma frase (em letras minúsculas) e imprime na tela a sequência de números que representa esta frase em um teclado não alfabético.
Esta função deve se chamar `transcrever_SMS` e ter o seguinte protótipo:

```
void transcrever_SMS(char frase[]);
```

Utilize o caracter 'x' para indicar uma pausa na digitação. Esta pausa é necessária para representar letras diferentes que são retratadas pelo mesmo número. Por exemplo, a frase "monalisa esta no louvre" corresponde à sequência "6x666x66255544477772033777782066x666055566688x88877733". Observe como o caracter 'x' separa a representação das letras 'm', 'o' e 'n', por exemplo.

Ao escrever esta função, pode-se assumir que a frase a ser processada não possui números, sinais de pontuação, acentos, caracteres maiúsculos etc.

Além disto, para facilitar a codificação, utilize as funções `codigo_letra` e `imprimir_codigos_letra`, a seguir.

```
int codigo_letra(char letra) {
    int codigo[]={2,2,2,3,3,3,4,4,4,5,5,5,6,6,6,7,7,7,7,8,8,8,9,9,9,9};
    if(letra == ' ') return 0;
    return codigo[letra-'a'];
}
```

A função `codigo_letra` recebe um caracter representando uma letra e devolve o seu código numérico. Desta maneira, após a execução da instrução `c = codigo_letra('b')`, por exemplo, a variável `c` armazenará o valor 2.

```
void imprimir_codigos_letra(char letra) {
    int rep[]={1,2,3,1,2,3,1,2,3,1,2,3,1,2,3,1,2,3,4,1,2,3,1,2,3,4};
    int i;
    if(letra == ' ') printf("%d", codigo_letra(letra));
    if(letra >= 'a' && letra <= 'z')
        for(i = 0; i < rep[letra-'a']; i++ )
            printf("%d", codigo_letra(letra));
}
```

A função `imprimir_codigos_letra`, por sua vez, recebe um caracter que representa uma letra entre 'a' e 'z' e já imprime na tela a sequência de dígitos correspondentes. Neste caso, por exemplo, após a execução da instrução `imprimir_codigos_letra('b')` será impresso "22" na tela.

```
void transcrever_SMS(char frase[]) {

}
```