	Universidade Federal de Campina Grande **Departamento de Sistemas e Computação** **Disciplina: Introdução à Programação – Turma: 3** **Período: 2019.2 – Prof. Roberto Faria** **26/11/2019**

Data Limite de Entrega: 03/12/2019

Capítulo 7 - Apontadores:

1. Escreva um programa que faça as seguintes ações:
   a) crie e inicialize um array com 20 elementos inteiros, na função ***main()***;
   b) crie um segundo array com 20 elementos inteiros, também na ***main()***;
   c) execute, a partir da ***main()***, uma função ***copia(array_original, tamanho, array_copia)***, do tipo ***void***, que receba os dois arrays criados e o tamanho, e, utilizando apontadores e sem usar indexação, copie os elementos do primeiro array para o segundo;
   d) Por último, imprima o array original e a cópia a partir da ***main()***.

2. Escreva um programa que receba dois conjuntos de inteiros, e os armazene em arrays. Os conjuntos poderão ter quantidades de elementos diferentes, mas, no máximo, cada um terá 50 elementos. A quantidade de elementos de cada conjunto também será informada pelo usuário. Utilize uma função ***uniao(primeiro_conjunto, tamanho_primeiro, segundo_conjunto, tamanho_segundo, conjunto_uniao, Tamanho_uniao)***, do tipo ***void***, para gerar um terceiro array com o conjunto união dos dois conjuntos recebidos. A função receberá os dois conjuntos recebidos na ***main()***, seus tamanhos, e, um conjunto vazio e seu tamanho (zero) para ser preenchido com o conjunto união e seu tamanho. Sua vida será mais fácil se você utilizar o método de procura sequencial, visto na sala de aula, para evitar as repetições. Não será utilizada indexação, apenas apontadores. O programa imprimirá os três conjuntos, a partir da função ***main()*** (Lembrete: por definição, um conjunto não tem elementos repetidos).

3. Faça um programa que leia vários inteiros positivos para um array de no máximo 100 elementos. O programa encerra a leitura dos elementos do array com uma sentinela zero ou negativa. Em seguida, o programa receberá ***n*** inteiros e, para cada um, imprimirá uma mensagem informando se o número está ou não presente no array. Para fazer a pesquisa, o programa utilizará o método da pesquisa binária, visto em sala, após a ordenação do array, também visto em sala. A função função de pesquisa será chamada assim: ***pesquisa_binaria(valor_procurado, array_de_procura, tamanho_do_array)*** e retornará um apontador para o elemento encontrado ou ***NULL*** (‘\0’), no caso do valor não ser encontrado. A função de ordenação será chamada assim:

bolha(array_de_procura, tamanho_do_array) e será do tipo ***void***. Não será utilizada indexação, apenas apontadores. Use funções independentes para ordenar e pesquisar no array.

4. Faça um programa que leia duas cadeia de caracteres, usando a função ***gets(cadeia)***, e concatene (junte) estas duas cadeias criando uma nova, com a primeira seguida da segunda. As cadeias terão no máximo 50 caracteres. A nova cadeia será armazenada num array de caracteres com 101 caracteres, usando a função ***concatena(primeira_cadeia, segunda_cadeia, array_para_concatenar)*** que recebe os endereços da duas cadeias e o endereço do array resultante. Não será utilizada indexação, apenas apontadores. Lembre-se: toda cadeia de caracteres no C termina com um caractere ***NULL*** ('\0').

5. Faça um programa que leia duas cadeias de caracteres, usando a função ***gets(cadeia)***, e mostre se a segunda cadeia está contida na primeira, através de uma mensagem. As cadeias terão no máximo 50 caracteres. A verificação será feita pela função ***contemcadeia(primeira_cadeia, segunda_cadeia)*** que recebe os endereços da duas cadeias e retorna verdadeiro (0) ou falso (diferente de 0). Não será utilizada indexação, apenas apontadores. Lembre-se: toda cadeia de caracteres no C termina com um caractere ***NULL*** ('\0').

6. Faça um programa que leia uma cadeia de caracteres, usando a função ***gets(cadeia)***, e mostre o número de ocorrências de cada letra do alfabeto nesta cadeia (não haverá distinção entre maiúsculas e minúsculas). A cadeia terá no máximo 250 caracteres. A contagem será feita por uma função ***conta_letras(cadeia, array_de_ocorrências)***, do tipo ***void***, que receberá a cadeia e um array de 26 elementos zerados para fazer a contagem das ocorrências de cada letra. A impressão será na função ***main()*** e não será utilizada indexação, apenas apontadores. Lembre-se: toda cadeia de caracteres no C termina com um caractere ***NULL*** ('\0').

7. Faça um programa que leia uma data num único inteiro no formato "ddmmaaaa" (por exemplo: 17041990) e mostre esta data por extenso (por exemplo: 17 de abril de 1990), utilizando um array de apontadores para cadeias de caracteres que contém os nomes dos meses por extenso (por exemplo, a declaração do array poderia ser: **`char *mes_extenso[13] = {"", "janeiro", "fevereiro", "março", "abril", "maio", "junho", "julho", "agosto", "setembro", "outubro", "novembro", "dezembro"};`** e o uso seria: **`mes_extenso[mes]`**, para obter o mês por extenso a partir do número do mês).

8. Faça um programa que mostre o numeral ordinal correspondente a um inteiro positivo, no intervalo de 1 a 1000, lido. Use arrays de apontadores para cadeias de caracteres e cadeias de caracteres para conter os ordinais da centena, dezena e unidade (por exemplo, a declaração do array para as unidades seria: **`char *unidade[10] = {"", "primeiro", "segundo", "terceiro", "quarto", "quinto", "sexto", "sétimo", "oitavo", "nono"};`**).

BOM TRABALHO!