Segunda Prova de Algoritmos e Estruturas de Dados I

Prof. Marcos Castilho

04 de fevereiro de 2003

Observações: A compreensão do enunciado faz parte da prova, que é individual e **sem** consulta e que, sendo um documento, deve ser feita à caneta. Qualquer fraude acarretará abertura de processo administrativo. Nos programas que você fizer será analisado: a lógica, o uso correto dos comandos, a sintaxe, a correta declaração dos tipos e os nomes das variáveis, a endentação, a clareza e a criatividade, além, é claro, do correto uso de procedimentos e funções, incluindo a correta passagem de parâmetros. **Esta prova tem três questões!**

1. (30 pontos)
   Dada uma sequência $x_1, x_2, \ldots, x_k$ de números reais, verifique se existem dois segmentos consecutivos iguais nesta sequência, isto é, se existem i e m tais que:

   $$x_i, x_{i+1}, \ldots, x_{i+m-1} = x_{i+m}, x_{i+m+1}, \ldots, x_{i+2m-1}.$$

   Imprima, caso existam, os valores de i e de m. Caso contrário, não imprima nada. Exemplo: Na sequência 7,9,5,4,5,4,8,6, existem $i = 3$ e $m = 2$.

2. (40 pontos)
   Um coeficiente binomial, geralmente denotado $\binom{n}{k}$, representa o número de possíveis combinações de n elementos tomados k a k. Um "Triângulo de Pascal", uma homenagem ao grande matemático Blaise Pascal, é uma tabela de valores de coeficientes combinatoriais para pequenos valores de n e k. Os números que não são mostrados na tabela têm valor zero. Este triângulo pode ser construído automaticamente usando-se uma propriedade conhecida dos coeficientes binomiais, denominada "fórmula da adição": $\binom{r}{k} = \binom{r-1}{k} + \binom{r-1}{k-1}$. Ou seja, cada elemento do triângulo é a soma de dois elementos da linha anterior, um da mesma coluna e um da coluna anterior. Veja um exemplo de um triângulo de Pascal com 7 linhas, com uma indicação de como obter os elementos:

1						
1	1					
1	2	1				
1	3	3	1			
1	4	6	4	1		
1	5	10	10	5	1	
1	6	15	20	15	6	1

   Faça um programa em PASCAL que imprima na tela um triângulo de Pascal com 10 linhas. Seu programa deve obrigatoriamente fazer uso de exatamente dois vetores durante o processo de construção. Um deles conterá a última linha impar gerada, enquanto que o outro conterá a última linha par gerada. Lembre-se que os elementos que não aparecem na tabela tem valor nulo. Você deve sempre ter o controle do tamanho da última linha impressa (o tamanho útil dos vetores em cada passo). Você deve também usar um procedimento para imprimir o vetor. Observe que não há entrada de dados, os dois vetores são gerados, um a partir do outro. O único elemento da primeira linha tem o valor 1. Você deve obrigatoriamente declarar um tipo VETOR com tamanho máximo TAM_MAX_VETOR, e o seu programa deverá tomar cuidado para manipular corretamente vetores de tamanho menor do que o tamanho máximo, impedindo que haja uma atribuição em posição ilegal de memória.

3. (30 pontos)

método de Newton é feita pelo refinamento desta solução inicial, isto é, pela tentativa de minimizar o erro cometido. Isto é feito pela expressão seguinte:

$$x_{n+1} = x_n - \frac{p(x_n)}{p\prime(x_n)},$$

$n = 0, 1, 2, \ldots$, e onde $p\prime(x)$ é a primeira derivada de $p(x)$. Usualmente, repete-se este refinamento até que $|x_{n+1} - x_n| < \epsilon, \epsilon > 0$, ou até que m iterações tenham sido executadas.

Construa um programa em PASCAL que receba como dados de entrada um polinômio $p(x) = a_0 + a_1x + a_2x^2 + \ldots + a_nx^n$ e uma aproximação inicial x_0 da raiz de $p(x)$, $\epsilon > 0$ e o número máximo de iterações, e calcule uma aproximação da raiz de $p(x)$ pelo método de Newton. Utilize obrigatoriamente um procedimento que receba como parâmetro um polinômio $p(x)$ (incluindo a informação sobre o grau do polinômio) e que calcule e retorne a função derivada $p\prime(x)$. Utilize também uma função que receba como parâmetros um polinômio $p(x)$ e um valor real $\overline{x}$ e retorne o valor do polinômio no ponto $\overline{x}$, isto é $p(\overline{x})$. Use esta função para calcular, a cada iteração do método de Newton, os valores de $p(x_n)$ e de $p\prime(x_n)$. Implemente o polinômio como sendo um vetor. Você pode usar o mesmo tipo VETOR e procedimento para imprimir do exercício 2. Você terá, no entanto, que implementar um procedimento para leitura, que não existe naquele exercício.