Algoritmia e Programação

Exame Época Especial – Parte Prática
5 de setembro de 2013

Duração: **2 horas e 10 minutos**

- **A interpretação do enunciado faz parte da resolução da prova. Se encontrar ambiguidades ou incoerências, resolva-as da melhor maneira e explique as decisões tomadas**

Grupo I

Cotação: 3 valores

Elabore um algoritmo que, dado um número inteiro positivo (**num**) com mais de 4 dígitos e um outro número inteiro positivo (**qtd**) maior que 1 e menor ou igual ao número máximo de dígitos de **num**, cria um novo número de acordo com a seguinte regra: os **qtd** dígitos mais à direita do número **num** são movidos, mas por ordem inversa, para o início de **num**.

Exemplo:
Se **num**=7834652 e **qtd**=4 então o número criado deve ser 2564783.

Nota: Deve apresentar todas as mensagens que sejam necessárias.

Grupo II

Cotações: 1- 2.5 val 2- 4 val 3- 1.5 val

Considere a seguinte classe com todas as funcionalidades descritas **devidamente implementadas:**

```
public class AcessoInstalacao {

   private static int lerAcessos (String nomeFx, String [][]info) throws FileNotFoundException{
   /* Lê os dados do ficheiro nomeFx e retorna a quantidade de elementos lidos.
Os dados do ficheiro são armazenados na matriz info, sendo os códigos dos utilizadores
armazenados na primeira coluna e as horas de acesso nas restantes 10 colunas). No entanto, as
horas não estão ordenadas e se o valor for "" (string vazia) quer dizer que não houve acesso.*/
   }
```

```
    private static int converteStringParaInt(String valor) {
        /* Converte a string valor com o formato "int1:int2" para um inteiro int1int2 e retorna-o.
Por exemplo o valor "9:03" é tranformado em 903.*/

    }

    private static void listarValores (String []dados)  {
        /* Lista para o ecrã os elementos  existentes no vetor dados. Cada elemento do vetor deverá
ter o formato "valor1;valor2". A escrita dos elementos é feita até encontrae um elemento com o
valor "fim" ou até chegar ao final do vetor.*/
    }

    public static void main(String[] args){
      /* ... A implementar ...  */
    }
}
```

Considere que:

- O formato aplicado à hora registada na matriz é "hora:minuto" e que está no formato 24h, isto é as horas vão desde 0:00 até às 23:59.

Implemente em JAVA os seguintes métodos:

1. **obterListaAcessosNoPeriodo**

   Recebe como parâmetros uma matriz de *strings*, a quantidade de elementos armazenados na matriz e uma hora de início e outra hora de fim. A matriz de *strings* tem na 1ª coluna o código do utilizador e nas restantes 10 colunas as horas de acesso (uma hora em cada coluna) ou "" (*string* vazia) se não houve acesso. Os parâmetros horas de início e fim devem ter o mesmo formato das horas de acesso registadas na matriz.

   O método deve retornar um vetor de *strings* com todos os acessos feitos entre os parâmetros hora de início e fim. Cada elemento do vetor, que representa os acessos, deve ter o seguinte formato: "código do utilizador;hora de acesso", exceto o último que deve ser "fim" (se necessário). Se um dado utilizador aceder mais que uma vez no intervalo compreendido entre os parâmetros hora de início e fim, cada acesso deve ser armazenado individualmente no vetor.

2. **contabilizarAcessosPorUtilizador**

   Recebe com parâmetro um vetor de *strings*, resultante do método **obterListaAcessosNoPeriodo** e retorna um vetor de *strings* com o seguinte formato "código do utilizador;quantidade de acessos". Em quantidade de acessos deve estar a quantidade de vezes que o código do utilizador aparece no vetor recebido como parâmetro. Na última posição do vetor retornado deve estar o valor "fim" (se necessário).

3. **main**

   Considerando todos os métodos da classe **AcessoInstalacao** corretamente implementados, incluindo os das alíneas anteriores, complete este método de forma a:

- Criar e preencher a matriz com os códigos dos utilizadores e as horas de acesso. Os dados devem ser lidos de um ficheiro indicado pelo utilizador. Assuma que no máximo haverá, no ficheiro, 50 utilizadores.
- Obter a lista de utilizadores que acederam entre as 9:30 e as 17:45.
- Apresentar a lista de acessos obtida.
- Apresentar a lista com a informação condensada, ou seja quantos acessos fez cada utilizador entre as 9:30 e as 17:45.

Grupo III

Cotações: 1- 4 val

Pretende-se criar um programa, em Java, que leia um ficheiro de texto (no formato apresentado abaixo à esquerda) com a listagem dos alunos de uma turma de APROG e cria um novo ficheiro (no formato apresentado abaixo à direita) que incluirá nota final de frequência obtida por cada aluno. **O novo ficheiro deve estar ordenado por ordem decrescente da nota final de frequência.** O programa deverá funcionar da seguinte forma:

1. Pedir ao utilizador para introduzir, via teclado, os nomes do ficheiro inicial (à esquerda) e do novo ficheiro (à direita);
2. Mostrar o conteúdo do ficheiro inicial, linha a linha, pedindo a cada passo as notas das 3 fichas de avaliação e a nota do dossiê obtidas pelo aluno;
3. O novo ficheiro incluirá, associado a cada aluno, a nota final de frequência obtida.
4. No final deve apresentar, no ecrã, uma mensagem com:
    a) A nota de frequência mais baixa e quantos alunos a obtiveram
    b) A nota de frequência mais alta e quantos alunos a obtiveram
    c) A média das notas de frequência da turma.

i112001; Joana Martins i112002;Francisco Pires i112003; Paulo Gomes

i112002 Francisco Pires 15 i112001 Joana Martins 13 i112003 Paulo Gomes 11