UNIVERSIDADE FERNANDO PESSOA

Disciplina: Programação **Ano:** 1º (época de trabalhadores) **Ano Lectivo:** 2002/2003
Exame: sem consulta; **Duração**: 2h 00m **Data:** 4 de Setembro de 2003
Licenciaturas: Engenharias (Ambiente, Civil, Comunicação, Informática)

1) Escreva um programa que leia do utilizador um número inteiro positivo e indique o número de ocorrências de cada algarismo (0-9) nesse número conforme apresentado no exemplo.

Exemplo:

Numero (n)? 1123

Ocorrências de algarismos em 1223: 0 aparece 0 vezes; 1 aparece 2 vezes; 2 aparece 1 vezes; 3 aparece 1 vezes; 4 aparece 0 vezes; 5 aparece 0 vezes; 6 aparece 0 vezes; 7 aparece 0 vezes; 8 aparece 0 vezes; 9 aparece 0 vezes;

1.1) Escreva o algoritmo em pseudo-código.
1.2) Escreva o respectivo algoritmo usando um fluxograma.
1.3) Faça o seguimento e teste (rastreio) do algoritmo para o número 1123.
1.4) Altere o pseudo-código de 1.1 de modo a que este contabilize o número de ocorrências de cada letra numa frase lida do utilizador (pode usar a função `ord(c)` para verificar qual o código ascii de cada caracter `c` da frase).
1.5) Altere o pseudo-código de 1.1 de modo que o resultado seja indicado na forma de um histograma conforme se vê em baixo:

Ocorrências de algarismos em 1223:

0:

1: **

2: *

.....

9:

(sugestão: use vectores para implementar os algoritmos pedidos)

2) Dada a seguinte declaração:

```
type
 pessoa = record
  nome:string[60]; morada:string[60]; telefone:string[9]; idade:integer;
 end;
var
 pessoas: array[1..100] of pessoa;
```

2.1) Escreva um procedimento que liste no ecrã o nome, morada, telefone e idade de todas as pessoas do vector `pessoas` com idades compreendidas entre dois valores `idade_min`, `idade_max` lidos do utilizador. Considere que o vector se encontra totalmente preenchido.

2.2) Altere o procedimento anterior para que liste no ecrã o nome, morada, telefone e idade de todas as pessoas que têm apelido 'Silva' no nome, que morem no 'Porto' e cujos últimos 4 dígitos do telefone estejam compreendidos entre 2500 e 7500.

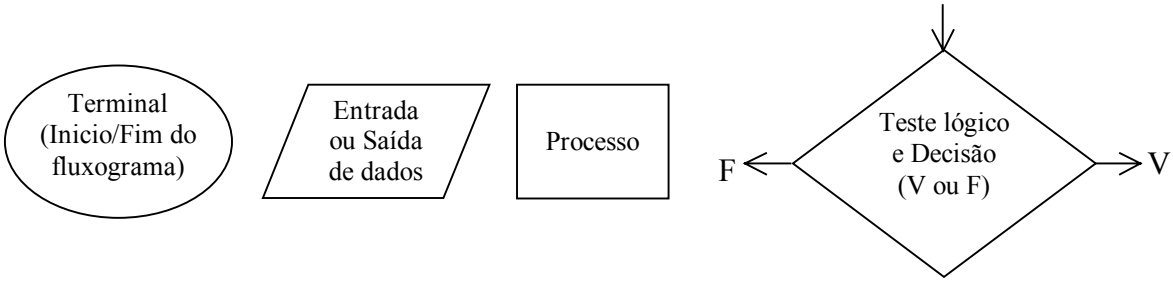
[Cotação em valores: 1.1) 3,5; 1.2) 2; 1.3) 2,5; 1.4) 2,5; 1.5) 2; 2.1) 2,5; 2.2) 5]

Sugestões de notação para descrição dos seus algoritmos ao longo do teste:

Pseudo-código:

- Leia(...) e LeiaLinha(...)
- Escreva(...) e EscrevaLinha(...)
- Para <*expressão_inicial*> Até <*expressão_final*> Faça <*bloco*>
- Se <*condição*> Então <*bloco*> Senão <*bloco*>
- Repita <*bloco*> AtéQue <*condição*>
- Enquanto <*condição*> Faça <*bloco*>
- Caso <*expressão*> Seja <*caso_1*> : <*bloco*> .... <*caso_n*> : <*bloco*> Senão <*bloco*> FimCaso
- <*operador de atribuição*>: ←
- <*bloco*>: Inicio .... Fim

Fluxogramas:



(Nota1: poderá, principalmente na descrição textual do algoritmo, usar outra notação com a qual esteja mais à vontade como por exemplo o Pascal ou outra. Não se preocupe demasiado com a sintaxe mas não deixe de ser claro na descrição do seu algoritmo
Nota2: poderá usar os operadores relacionais, lógicos e aritméticos normais assim como algumas funções matemáticas mais vulgares)

Vectores e Cadeias de Caracteres:

Considere os vectores e matrizes como sendo de tamanho fixo
Relativamente às variáveis do tipo *string* considere o seguinte:

- Pode consultar/alterar o valor de um caracter dentro da string através da operação normal de acesso a elementos de um vector
- Operador de concatenação de strings: +
- Função que devolve o comprimento da string: length(*s*)
- Função para copiar/extrair substring de uma string: copy (*s, index_ini, count*)
- Função para determinar posição de uma substring dentro de uma string: pos(*substr, s*)
- Conversão de string para um valor numérico: val(*s, v, error_code*)
- Conversão de um valor numérico para string: str(*x, s*)