Nome do estudante: _____ **Código:** _____

1. [7.0 pontos = 3.0 + 2.0 + 2.0]

Pretende-se desenvolver um programa para processar os nomes de um conjunto de pessoas contidos num ficheiro de texto, guardando os resultados do processamento noutro ficheiro de texto. O processamento consiste em simplicar o nome original de cada pessoa, de modo a que o nome simplificado tenha apenas a primeira e a última palavra (se existente) do nome original. Em ambos os ficheiros, cada nome ocupa uma linha de texto (ver exemplo abaixo).

a) [3.0] Considerando que lhe é fornecida uma função **void simplify(const string &name, string &sName)** que converte um nome original, **name**, num nome simplificado, **sName**, escreva o código completo de um programa que faz o seguinte:

- abre o ficheiro **people1.txt** para leitura e o ficheiro **people2.txt** para escrita; o primeiro contém os nomes originais; no segundo serão guardados os nomes simplificados; se não for possível abrir com sucesso o ficheiro **people1.txt** o programa deve escrever a mensagem "people1.txt not found" na saída de erro padrão e terminar imediatamente com um *exit code* igual a 1; considere que o ficheiro **people2.txt** é sempre criado com sucesso;
- lê o nome de cada pessoa contido no ficheiro **people1.txt**, simplifica o nome, usando a função **simplify**, e escreve o nome simplificado no ficheiro **people2.txt**.

NOTAS: 1) Indique as *header files* a incluir; 2) Não implemente ainda a função **simplify**.

Exemplo de conteúdo do ficheiro **people1.txt**:

```
Franclim da Horta e Jardim
Maria
Salvador Socorro
Caio Passos Dias Aguiar Mota
...
```

Exemplo de conteúdo do ficheiro **people2.txt**:

```
Franclim Jardim
Maria
Salvador Socorro
Caio Mota
...
```

b) [2.0] Escreva o código da função **simplify** cujo protótipo foi definido na questão anterior. Considere que o nome original pode ter um número de palavras variável, em particular, apenas uma palavra (ex: Maria) e não tem espaços antes da primeira nem depois da última palavra.

c) [2.0] Indique as alterações a introduzir no programa que escreveu em **1.a)**, de modo a que os nomes simplificados sejam escritos no ficheiro **people2.txt** por ordem alfabética. Indique apenas as alterações, assinalando o seu local no código que escreveu em **a)**, usando letras maiúsculas (**A**,**B**,**C**, ..) dentro de um círculo (exemplo: Ⓒ); use os mesmos símbolos no código que apresentar abaixo. Indique também as alterações nas *header files*.

Nome do estudante: _____ **Código:** _____

2. [5.0 pontos = 1.0 + 1.2 + 1.3 + 1.5]

Considere a declaração parcial da classe **Date**, usada para representar datas (ano, mês e dia):

```
class Date
{
  friend istream & operator>>(istream & f, Date & date);
  friend ostream & operator<<(ostream & f, const Date & date);
  friend bool operator<(const Date& left, const Date& right);
  friend bool operator==(const Date& left, const Date& right);
public:
  Date(int y=1, int m=1, int d=1);
  // ... other methods
private:
  int y, m, d; // year, month, day
};
```

Exemplo de execução de um programa que usa a classe **Date** *(a desenvolver na questão **2.d**):*

```
Date1 (yyyy:mm:dd)? 2017-6-13
Date2 (yyyy:mm:dd)? 2000-1-1
2017-6-13 is after 2000-1-1
```

a) [1.0] Escreva o código do construtor da classe **Date**. Considere que os parâmetros do construtor representam uma data válida.

b) [1.2] Escreva o código do **operator<** que determina se a data **left** é anterior à data **right** (*ver protótipo acima*).

SUGESTÃO: construa dois números inteiros ou duas *strings* que representem as datas (por exemplo, a data 2017-6-13 pode ser representada por 20170613 e a data 2000-1-1 por 20000101) e compare os seus valores.

c) [1.3] Escreva o código da função que faz a sobrecarga (*overloading*) do **operator>>** para a classe **Date**. Se a data não for inserida no formato correto, isto é, usando um hífen para separar o dia do mês e outro hífen para separar o mês do ano (*ver exemplo na página anterior*) ou se não forem inseridos valores numéricos inteiros positivos para o dia, o mês e o ano, esta função deve "lançar uma exceção" (*throw an exception*), "arremessando" um objeto do tipo **runtime_error**, contendo a mensagem "**invalid Date**". NOTA: por simplificação, considere que a função a desenvolver apenas testa se a data está num formato correto e não testa se é uma data válida.

d) [1.5] Escreva a função **main** de um programa que lê duas datas, compara-as e mostra, no ecrã, uma mensagem indicando se a primeira data é anterior ("before"), igual ("equal") ou posterior ("after") à segunda data (*ver exemplo junto da definição da classe* **Date**). Considere que todos os métodos e funções *friend* da classe **Date** apresentados na declaração da classe, na página anterior, estão implementados. O programa deve fazer o tratamento das eventuais exceções geradas durante a leitura de uma data (*ver questão anterior*), apresentando no ecrã a mensagem enviada com o objeto "arremessado" e terminando imediatamente com um *exit code* igual a 1.

Nome do estudante: _____ **Código:** _____

3. [5.0 pontos = 0.5 + 1.5 + 1.0 + 1.5 + 0.5]

Uma imagem é representada por uma matriz bidimensional; cada elemento da matriz representa o valor de um pixel. Considere a declaração parcial de uma classe **Image**:

```
class Image {
public:
  Image(size_t nLins=0, size_t nCols=0, int pixVal=0); //nLins & nCols are the dimensions →
                                                       // → of the image; pixVal is the initial value of every pixel
  void setPixel(size_t lin, size_t col, int pixVal); //modifies the value of the pixel at (lin,col)
  bool read(string fileName); // reads the image saved in file fileName, storing it in img ;→
                                                                  // → returns true if sucessful, false otherwise
  Image getRegion(size_t lin, size_t col, size_t nLins, size_t nCols) const;
                              // returns an image containing a rectangular region of img
  // ... other methods
private:
  vector < vector<int> > img; // image representation
  // ... other attributes
};
```

a) [0.5] Complete a declaração da classe, escrevendo o protótipo (*apenas o protótipo*) do método **getPixel** que retorna o valor de um pixel da imagem cujas coordenadas (linha e coluna) recebe como parâmetros.

b) [1.5] Escreva o código do método **getRegion(size_t lin, size_t col, size_t nLins, size_t nCols)** que retorna uma (sub-)imagem que representa uma região/porção retangular da imagem. Os parâmetros **lin** e **col** são as coordenadas (linha e coluna) do canto superior esquerdo da região selecionada; **nLins** e **nCols** são as dimensões (número de linhas e número de colunas) dessa região. Considere que todos os parâmetros tomam valores válidos.

c) [1.0] Pretende-se desenvolver um programa que guarde, numa estrutura de dados, em memória principal, um conjunto de imagens organizadas por datas, permitindo aceder de forma eficiente às imagens adquiridas numa determinada data. A uma data podem estar associadas várias imagens. As imagens são representadas por objetos da classe **Image**; as datas são representadas por objetos da classe **Date** (ver questão **2**). Declare a estrutura de dados que usaria para representar a referida informação.

Justifique, brevemente, a sua escolha.

d) [1.5] Escreva a parte do código do referido programa que, usando as classes **Date** e **Image** e a estrutura de dados escolhida em **c)**, faz o seguinte: **1)** lê uma data; **2)** lê do teclado, um por um, os nomes dos ficheiros que contêm as imagens adquiridas nessa data (as entradas terminam quando for lida uma *string* vazia) e **3)** lê as imagens contidas nos ficheiros indicados e insere-as na referida estrutura de dados, associando-as à data lida. Use os métodos declarados nas classes **Date** e **Image**; considere que estão todos implementados. Considere também que todas as imagens são lidas com sucesso.

e) [0.5] Na classe **Image** declarada anteriormente os valores dos pixels são números inteiros. Indique as alterações a introduzir na declaração da classe **Image** de modo e transformá-la numa template class que possa ser usada para representar imagens com outros tipos de pixeis, por exemplo **float**, **double** ou **char**. Indique apenas as modificações.

Nome do estudante: _____ **Código:** _____

4. [3.0 valores = 0.5 + 0.5 + 0.5 + 0.5 + 0.5 + 0.5]

a) [0.5] A função **readNumber** (abaixo) mostra a mensagem **msg** que recebe como parâmetro e retorna/devolve o número lido do teclado. Complete a função, de modo a que tenha dois protótipos diferentes. Em ambos os casos, indique como faria uma chamada à função para mostrar a mensagem **"Grade ? "** e ler um valor, guardando-o na variável **grade**.

`_____ readNumber(_____)`
`{`
`  ...`
`  cout << msg; cin >> number;`
`  ...`
`}`
	`_____ readNumber(_____)`
	`{`
	`  ...`
	`  cout << msg; cin >> number;`
	`  ...`
	`}`

chamada:	chamada:

b) [0.5] Considere a classe **Date** (da questão **2**). É possível fazer a seguinte declaração: **Date date1;** ? Justifique.

c) [0.5] Admita que não tinha acesso aos contentores da **STL** (Standard Template Library), nomeadamente **vector**'s. Nessa situação, faça a declaração dos atributos que usaria na classe **Image** da questão **3**.

Tendo em conta a nova forma de guardar uma imagem, seria necessário acrescentar algum método à classe **Image**? Justifique.

d) [0.5] Implemente o construtor da classe **Image** na situação descrita na questão anterior (**4.c**).

e) [0.5] A STL de C++ disponibiliza uma função (algoritmo) **sort()** cujo *template* é o seguinte:

```
template <class RandomAccessIterator>
 void sort (RandomAccessIterator first, RandomAccessIterator last);
```

Explique por que não é possível usar esta função para ordenar os elementos de um contentor do tipo **list**, da STL.

f) [0.5] Um desenho é constituído por múltiplos objetos de diferentes tipos: retângulos, triângulos, e circumferências. Para representá-los, um programador definiu as classes **Rectangle**, **Triangle** e **Circle**, derivadas de uma classe base, **Shape**. A classe **Shape** tem um método público cujo protótipo é **virtual void draw() = 0;** . Qual o significado e implicações desta definição do método **draw**?

FIM