

Unidade Curricular

Programação Imperativa
Laboratórios de Informática

Projeto de avaliação

Licenciatura em Engenharia e Sistemas Informáticos

EST / IPCA

2025/2026

Índice

Introdução	2
Sistema de Faturação e Geração de Ficheiro de Débitos Diretos (Formato PS2)	2
1. Objetivos	2
2. Descrição do Problema	2
3. Ficheiros de Entrada	3
4. Funcionalidades a Implementar	3
5. Descrição do Formato PS2 Simplificado	4
6. Requisitos Técnicos e Constrangimentos	4
7. Deliverables e avaliação	4
Anexo A: Algoritmo de Validação de NIF	5
Anexo B: Algoritmo de Validação de NIB	5

Introdução

A realização do projeto tem como objetivo (i) consolidar a aprendizagem dos assuntos leccionados e (ii) exercitar os conhecimentos e técnicas adquiridos através da sua aplicação numa situação real.

Observações

As submissões dos projetos de avaliação deverão ser efetuadas através da plataforma Moodle até ao dia 5 de janeiro de 2026. Deverá ser submetido um ficheiro ZIP com todo o material desenvolvido. Os projetos de avaliação são de realização em grupo (com um máximo de 4 elementos).

A defesa oral de cada projeto é obrigatória, realizando-se em data a divulgar posteriormente.

Não é permitida a realização (ou melhoria) da componente prática em: época de exames; época especial ou época excecional.

Os critérios de avaliação incorporam os pontos seguintes:

- Qualidade do código desenvolvido (15%);
- Qualidade da solução desenvolvida (15%);
- Qualidade da documentação produzida (10%);
- Qualidade da defesa do projeto desenvolvido (60%).

Sistema de Faturação e Geração de Ficheiro de Débitos Diretos (Formato PS2)

1. Objetivos

O presente trabalho prático tem como principal objetivo a aplicação dos conceitos fundamentais de programação imperativa na resolução de um problema do mundo real. Pretende-se que os estudantes desenvolvam uma aplicação em linguagem C capaz de processar dados de faturação de uma empresa de serviços (e.g., eletricidade) e gerar um ficheiro para cobrança por débito direto, de acordo com um layout normalizado.

Competências a desenvolver:

- Manipulação de tipos de dados estruturados (**structs**) para modelar entidades do problema (Clientes, Consumos, etc.).
- Utilização de **arrays** para armazenar e gerir coleções de dados em memória.
- Manipulação de ficheiros de texto para leitura de dados de entrada e escrita de dados de saída.
- Implementação de algoritmos de validação de dados (NIF e NIB).
- Desenvolvimento de código modular, organizado e devidamente comentado.
- Compreensão e implementação de um formato de ficheiro com layout fixo (PS2 simplificado).

2. Descrição do Problema

A empresa "IPCA Energy" fornece eletricidade aos seus clientes e pretende automatizar o processo de cobrança mensal. A empresa dispõe de ficheiros de texto com a informação dos seus clientes, os consumos mensais registados e o preço em vigor para cada período.

O sistema a desenvolver deverá ler estes ficheiros, processar a informação, calcular o valor a faturar a cada cliente e, por fim, gerar um único ficheiro de texto no **formato PS2 (simplificado)**. Este ficheiro será posteriormente enviado ao banco para efetuar a cobrança automática na conta dos clientes que têm o serviço de débito direto ativo.

Uma etapa crucial do processo é a **validação dos dados dos clientes**. O sistema deve verificar se o NIF e o NIB de cada cliente são válidos antes de o incluir no ficheiro de débitos diretos. Clientes com dados inválidos deverão ser ignorados

no ficheiro final e reportados numa listagem de erros.

3. Ficheiros de Entrada

A aplicação deverá ler os dados a partir dos seguintes ficheiros de texto, cujo formato é `campo1;campo2;...;campoN`.

- **clientes.txt:** Armazena a informação dos clientes da empresa.
 - **Formato:** ID_Cliente;Nome;NIF;NIB

Exemplo:

1001;Joao Silva;235689142;003506510000258741254
1002;Maria Sousa;501845923;003300004521087459615
1003;Carlos Pereira;123456789;001000001234567890123

◦

- **consumos.txt:** Regista os consumos (em kWh) de cada cliente por período.
 - **Formato:** ID_Cliente;Ano;Mes;Consumo_kWh

Exemplo:

1001;2025;10;150
1002;2025;10;275
1003;2025;10;85
1001;2025;09;142

◦

- **periodos.txt:** Define o custo por kWh para cada período de faturação.
 - **Formato:** Ano;Mes;Preco_kWh

Exemplo:

2025;09;0.152
2025;10;0.165
2025;11;0.168

◦

4. Funcionalidades a Implementar

O programa deverá solicitar ao utilizador o **ano** e o **mês** para o qual se pretende gerar o ficheiro de cobrança. Com base nessa informação, deverá executar as seguintes tarefas:

1. **Carregamento de Dados:** Ler o conteúdo dos três ficheiros de entrada para arrays de structs apropriadas (e.g., struct Cliente, struct Consumo, struct Período).
2. **Validação de Dados:** Para cada cliente lido:
 - Implementar e aplicar o algoritmo de **validação do NIF** (descrito no Anexo A).
 - Implementar e aplicar o algoritmo de **validação do NIB** (descrito no Anexo B).
 - Clientes com NIF ou NIB inválidos não devem ser incluídos no ficheiro de débitos. Uma mensagem de erro deve ser apresentada no ecrã para cada cliente inválido, indicando o ID do cliente e o motivo da invalidade.
3. **Processamento de Faturação:**
 - Determinar o preço por kWh para o período (ano/mês) indicado pelo utilizador, consultando os dados carregados de periodos.txt.
 - Para cada cliente válido, encontrar o seu consumo correspondente ao período.
 - Calcular o valor a debitar: $Valor = Consumo_kWh * Preco_kWh$.
4. **Geração do Ficheiro de Débitos Diretos (PS2):**

- Criar um ficheiro de texto com o nome DD_AAAA_MM.ps2 (ex: DD_2025_10.ps2).
- Escrever o conteúdo no ficheiro de acordo com a especificação do **Formato PS2 Simplificado** (descrita na Secção 5).
- O ficheiro deve conter um registo de cabeçalho, vários registos de detalhe (um por cliente a ser debitado) e um registo de fim.
- Os totais no cabeçalho e no registo de fim (valor total e número de débitos) devem ser calculados dinamicamente durante o processamento.

5. Descrição do Formato PS2 Simplificado

O formato PS2 é um *standard* de comunicação de ficheiros entre empresas e bancos. Para este trabalho, será utilizada uma versão simplificada com um **layout de largura fixa**. Cada linha do ficheiro é um registo e cada campo dentro do registo tem uma posição e um tamanho fixos (ver ficheiro *Layout_PS2.pdf*):

- Registo de Cabeçalho (Tipo 1): A primeira linha do ficheiro. Contém informação geral sobre o ficheiro, como a empresa emissora, a data de processamento, e os totais (valor total e número de operações).
- Registo de Detalhe (Tipo 2): Uma linha por cada cliente a debitar. Cada linha corresponde a uma operação individual (um débito a um cliente ou um crédito a um beneficiário).
- Registo de Fim (Tipo 9): A última linha do ficheiro. Serve como um fecho e repete os totais do cabeçalho para validação e controlo.

Exemplo de um ficheiro DD_2025_10.ps2:

```
120251026IPCA Energy          50123456700000000045375000002
2000000100100350651000025874125423568914200000000024750Fatura Eletricidade 10/2025
2000000100200330000452108745961550184592300000000020625Fatura Eletricidade 10/2025
90000000000453750000002
```

6. Requisitos Técnicos e Constrangimentos

- A linguagem de programação a utilizar é o C.
- É **obrigatória** a utilização de structs e arrays.
- A utilização de alocação dinâmica de memória (malloc, calloc, etc.) **não é permitida**. Os arrays devem ter uma dimensão estática, suficientemente grande para os dados de exemplo.
- O código deve ser modular, ou seja, bem organizado em funções (e.g., carregarClientes(), validarNIF(), gerarFicheiroPS2(), etc.).
- O código fonte deve ser devidamente comentado para facilitar a sua compreensão.

7. Deliverables e avaliação

- **A entregar:** Um ficheiro ZIP contendo:
 - Todos os ficheiros de código fonte (.c e .h).
 - Um ficheiro relatorio.pdf (máximo 2 páginas) descrevendo a estrutura do programa, as opções tomadas e as dificuldades encontradas.
 - Os ficheiros de entrada utilizados para os testes.
 - Um exemplo do ficheiro DD_AAAA_MM.ps2 gerado pela aplicação.
- **Critérios de Avaliação:**
 - **Funcionalidade (40%):** O programa compila sem erros e executa todas as funcionalidades pedidas corretamente.
 - **Validações e Lógica (30%):** A validação de NIF/NIB e o cálculo de faturação estão corretos. O ficheiro PS2 é gerado no formato exato.
 - **Qualidade do Código (20%):** Legibilidade, organização, modularidade, comentários e utilização correta de structs e arrays.

- **Relatório (10%):** Clareza e pertinência da documentação entregue.

Anexo A: Algoritmo de Validação de NIF

Um NIF (Número de Identificação Fiscal) português válido tem 9 dígitos. O primeiro dígito deve pertencer a um conjunto específico (1, 2, 5, 6, 8, ou 9). O nono dígito é um dígito de controlo.

Algoritmo:

1. Verificar se o NIF tem 9 dígitos e se o primeiro dígito é válido.
2. Calcular o dígito de controlo:
 - Multiplicar o 1º dígito por 9, o 2º por 8, ..., o 8º por 2.
 - Somar todos os resultados.
 - Calcular o resto da divisão desta soma por 11 (*Soma mod 11*).
 - Se o resto for 0 ou 1, o dígito de controlo é **0**.
 - Caso contrário, o dígito de controlo é **11 - Resto**.
3. Comparar o dígito de controlo calculado com o 9º dígito do NIF. Se forem iguais, o NIF é válido.

Exemplo (NIF 501845923):

- $Soma = (5 * 9) + (0 * 8) + (1 * 7) + (8 * 6) + (4 * 5) + (5 * 4) + (9 * 3) + (2 * 2) = 171$
- $Resto = 171 \bmod 11 = 6$
- Dígito de controlo = $11 - 6 = 5$. (Inválido, o último dígito é 3. Este NIF seria rejeitado).

Anexo B: Algoritmo de Validação de NIB

O NIB (Número de Identificação Bancária) é composto por 21 dígitos: 4 para o banco, 4 para o balcão, 11 para o número de conta e 2 dígitos de controlo. A validação usa o algoritmo de módulo 97.

Nota: Embora o NIB esteja a ser substituído pelo IBAN (PT50 + NIB), a validação dos 21 dígitos do NIB continua a ser um excelente exercício de programação.

Algoritmo:

1. Verificar se o NIB tem 21 dígitos numéricos.
2. Calcular o módulo 97 de todo o número de 21 dígitos.
3. Se o resultado for **1**, o NIB é considerado válido.

Como um número de 21 dígitos excede a capacidade dos tipos de dados inteiros padrão, este cálculo deve ser feito por partes ou tratando o número como uma string de caracteres. Uma abordagem comum é processar os dígitos em blocos.

Exemplo (NIB 003506510000258741254):

- $003506510000258741254 \bmod 97 = 1$. O NIB é válido.