Universidade da Beira Interior
Departamento de Informática

Team nº 2 - 2020: *Help is on the way*

Elaborado por:

Gonçalo Valente
Marco Bernardes
Pedro Hilário
Rúben Ferraz
Ruben Serrano

Orientador:

Professor Doutor Pedro R. M. Inacio

1 de fevereiro de 2021

Agradecimentos

A realização do presente relatório não seria possível sem o apoio, contributo e esforço de várias pessoas que direta ou indiretamente influenciaram o rumo deste trabalho. Assim sendo, pretendemos agradecer a todos que sempre nos encorajaram a seguir em frente e a fazer sempre melhor.

Ao professor Pedro Ricardo Morais Inácio, pela sua orientação, total colaboração e disponibilidade, pelos seus ensinamentos que contribuem para o nosso enriquecimento educacional, por toda a ajuda prestada, não apenas neste trabalho, mas fundamentalmente ao longo de todo o semestre, sendo um privilégio e uma honra sermos seus educandos. A ele, o nosso muito Obrigado!

A todos os docentes que contribuem para a nossa formação e crescimento académico ao longo desta licenciatura, pelo saber que transmitem, os valiosos conhecimentos que nos incutem e pela excelência da qualidade técnica de cada um.

Aos nossos colegas de curso pela amizade e solidariedade, sempre disponíveis para nos ajudar, aconselhar e encorajar nos momentos cruciais desta caminhada estudantil.

Aos companheiros com os quais desenvolvemos este trabalho, pela dedicação, empenho inexcedível e saudavelmente exigível, pela partilha do esforço e dos inúmeros desafios que enfrentamos, sempre com o espírito colaborativo.

E por fim, cada elemento do grupo, agradece de forma especial à sua família, pela confiança no nosso progresso, pelo apoio incondicional e por nos incentivarem e apoiarem em todas as áreas da nossa vida.

A todos os nosso sincero e profundo Muito obrigado!

Conteúdo

Lista de Excertos de Código

Acrónimos

UC	Unidade Curricular
UC's	Unidades Curriculares
ALGA	Álgebra Linear e Geometria Analítica
LP	Laboratórios de Programação
VSCode	Visual Studio Code

Capítulo

1

Introdução

1.1 Âmbito, Enquadramento e Motivação

O aumento cada vez mais significativo e essencial das novas tecnologias de informação com o objetivo de uma utilização através de meios mais expeditos, viáveis e eficazes, leva a que se implemente programas que facilitem ao utilizador o uso desses meios de uma forma prática e rápida!

O presente trabalho consiste no desenvolvimento de um programa com novas funcionalidades que responda a essas necessidades, ou seja, na criação e implementação de um programa com tais características.

Com base nos conhecimentos provenientes de outras Unidades Curriculares (UC's), mais concretamente Programação e Álgebra Linear e Geometria Analítica (ALGA) decidiu-se começar a trabalhar neste projeto.

Este trabalho que consiste na implementação de um programa, que facilite a execução de uma forma prática e fácil, dos dados (informação) que o utilizador pretende frequentemente utilizar, enquadra-se no âmbito da Unidade Curricular (UC) de Laboratórios de Programação (LP), baseado na necessidade de criação de programas que analisem, organizem, calculem e filtrem a informação necessária e útil.

1.2 Motivação

Com o avanço tecnológico urge a necessidade de criar programas com ferramentas de execução rápida e fácil e este trabalho surge como uma resposta a essa necessidade, pois demonstra o desenvolvimento de um programa que armazena e administra dados específicos que o utilizador aplica habitualmente.

1.3 Objetivos

No âmbito da UC de LP elaborámos o presente trabalho prático, tendo como principal objetivo, implementar um programa que solicitasse ao utilizador 16 números inteiros entre 6 e 14 e que os guardasse num vetor, para posteriormente providenciar forma de calcular algumas estatísticas ou fazer operações sobre esses valores, tais como:

1. A identificação do valor mais próximo de 15;
2. O cálculo da multiplicação de cada elemento do vetor pelo último elemento do vetor em causa;
3. A construção de uma matriz 16 por 16 em que a sua primeira linha é composta pelo vetor lido e as restantes constituídas por permutações dos seus valores;
4. A devolução dos valores do vetor que são divisíveis por três;
5. O cálculo do seno de todos os elementos do vetor;
6. E por fim, a devolução do vetor ordenado por ordem crescente.

Procedendo desta forma, pretende-se guardar informação útil e necessária ao utilizador, que a poderá utilizar futuramente de uma forma mais eficaz, célere e prática! Para a realização do presente trabalho e com o objetivo de enriquecimento do mesmo, foi proposto a opção de incluir funcionalidades adicionais, e no cumprimento dessa proposta, apresentamos as seguintes funcionalidades:

1. A leitura de um novo vetor, cálculo e devolução do seu produto interno;
2. A identificação de todos os números primos no vetor inicial;
3. A leitura de um novo vetor um por 16, cálculo e devolução da matriz 16 por 16 resultante do produto do vetor inicial com o novo vetor gerado;
4. O cálculo do determinante da matriz referida no ponto anterior;
5. A presença de uma página de ajuda, sendo a sua entrada o algarismo sete no menu;
6. Para concluir o programa mostra algum tipo de ajuda quando é executado a partir da linha de comandos com a flag –help.

1.4 Abordagem

Na abordagem ao trabalho a realizar, começou-se por uma reunião dos elementos do grupo, a fim de se decidir qual o programa a utilizar, que métodos escolher e também selecionar sites e fóruns de pesquisa que nos fornecesse informação útil e adequada ao pretendido.

Para colmatar certas dúvidas que foram surgindo ao longo da elaboração do trabalho, houve necessidade de recorrer ao superior conhecimento do professor da disciplina, para nos orientar e esclarecer de forma a haver uma melhor prestação nas tarefas a realizar.

A organização do trabalho passou por uma planificação das tarefas e tópicos a realizar, de forma a tornar o desempenho mais produtivo e eficaz. Após planificação, foi distribuída aos vários elementos do grupo, os tópicos que cada um especificamente devia trabalhar, e consequentemente as tarefas a executar para o cumprimento desses tópicos. Essa distribuição baseou-se em certos critérios, nomeadamente o talento, aptidão e conhecimento de cada um, para o tópico e tarefa a atribuir.

Foram ainda definidos objetivos a atingir e a desempenhar para cada um dos membros do grupo. O mesmo reuniu frequentemente para partilhar e dar a conhecer o que já estava elaborado, e se assim se entendesse relevante e oportuno, proceder a alterações ou correções.

1.5 Organização do Documento

De modo a refletir o trabalho que foi feito, este documento encontra-se estruturado da seguinte forma:

1. O capítulo 1 – **Introdução** – apresenta o projeto, a motivação para a sua escolha, o enquadramento para o mesmo, os seus objetivos e a respetiva organização do documento.

2. O capítulo 2 – **Estado da Arte** – descreve o panorama geral das tecnologias e conhecimentos utilizados durante a realização deste trabalho e respetivo enquadramento no trabalho realizado.

3. O capítulo 3 – **Tecnologias e Ferramentas Utilizadas** – enumera as plataformas utilizadas na edição do código, na criação do presente relatório, em LaTeXe na criação da documentação do programa criado.

4. O capítulo 4– **Implementação e Testes** – onde são descrios os passos para a implementação dos algoritmos utilizados, bem como a sua explicação.

5. O capítulo 5 – **Conclusões e Trabalho Futuro** – onde são apresentadas as conclusões dos elementos do grupo, após a realização do trabalho e os possíveis trabalhos a serem realizados no futuro.

Capítulo

2

Estado da Arte

2.1 Introdução

Neste capítulo vão ser enumerados e explicados, individualmente, cada um dos *headers* introduzidos neste trabalho, assim como também algumas das ferramentas que estão disponibilizadas online com funções idênticas ao mesmo.

Por fim irá ser feita uma breve conclusão acerca do capítulo em causa.

2.2 Estado da Arte

Para a realização deste trabalho foram usadas algumas das bibliotecas (também conhecidas por *headers*) pertencentes à linguagem C, como: `<stdio.h>`; `<stdlib.h>`; `<string.h>`; `<time.h>`; `<ctype.h>`. Programou-se especialmente para este trabalho uma biblioteca, `"functionalities.h"`, com o intuito de satisfazer as funcionalidades pedidas no enunciado (não só as mínimas mas também as mais elaboradas).

Cada uma destas bibliotecas tem uma função distinta, sendo que o trabalho não teria sido possível se não estivessem incluídas.

Para começar tem-se o *header* `<stdio.h>`, que tem como função melhorar o desempenho do *input* e do *output*, respetivamente; a `<stdlib.h>` que melhora as funções em geral; a `<string.h>` que permite a criação de *arrays* de caracteres; a `<time.h>` que foi usada para poder manipular o tempo e data; por fim a `<ctype.h>` que declara algumas funções que são úteis para testar e mapear caracteres.

2.3 Trabalhos Relacionados

Relacionado com o tema deste trabalho existem várias ferramentas *online*, sendo que as que mais se interligam com o mesmo são o *Symbolab Math Solver*, o *Wolfram Alpha* e o *Matrixcalc*. Todos eles permitem a manipulação de matrizes, assim como outras operações matemáticas.

Começando pelo *Matrixcalc*, este é um programa que permite realizar operações entre matrizes, resolver sistemas de equações lineares, calcular o determinante de uma matriz, entre outras. Contém ainda uma secção com alguma teoria para facilitar a pesquisa e a realização dos cálculos.

De seguida tem-se o *Wolfram* que proporciona ao utilizador uma completa lista de tópicos, como matemática, ciência e tecnologia, sociedade e cultura, entre outros. É uma ferramenta bastante completa e de fácil utilização, sendo que a maioria dos resultados tem uma resolução *step-by-step*.

Por fim, e não menos importante, inclui-se o *Symbolab*. Esta ferramenta permite, à semelhança das referidas anteriormente, a resolução de diversos problemas na área da matemática assim como inclui uma secção chamada *cheat sheets* que contém uma vasta diversidade de fórmulas e regras matemáticas.

2.4 Conclusões

Inicialmente foi enunciado no ponto 2.1 do que se vinha a tratar este capítulo, sendo que foram mencionados os *headers* e consequentemente a função de cada um deles. Sequencialmente foram referidos alguns programas relacionados com o trabalho em causa, como o *Symbolab Math Solver*, o *Wolfram Alpha* e o *Matrixcalc*, bem como as suas funcionalidades.

Para finalizar pode-se concluir que nos dias de hoje existem inúmeros programas que facilitam temas como o tratado aqui, manipulação de matrizes, em que o nosso trabalho se inclui.

Capítulo

3

Tecnologias e Ferramentas Utilizadas

3.1 Introdução

Neste capítulo pretendemos explicar os motivos de decisão dos programas escolhidos, explanando os seus atributos e vantagens para o objetivo pretendido.

3.2 Secções Intermédias

Para a concepção do trabalho recorremos a três programas essenciais sendo estes: o *Overleaf* , o *Visual Studio Code (VSCode)* e o *Doxygen*.

Optamos por escolher o *Overleaf* para a realização do relatório , pois o mesmo traz várias vantagens, tais como:

1. O padrão matemático em LaTeX concebe funções e equações perfeitamente formatadas;
2. Podemos implementar comentários no mesmo espaço em que o conteúdo é procriado;
3. Em LaTeX a linguagem é completa pois podemos formar funções que nos traz mais facilidade;
4. Não existe incompatibilidade entre versões;
5. LaTeX é o modelo mais usado em diversas áreas de conhecimento;

6. Ao contrário de outros programas o LaTeX é gratuito.

Utilizámos o *VSCode* para implementar o código trazendo este as seguintes vantagens:

1. É um programa que pode ser utilizado em vários tipos de *software*, (*Linux*, *Windows*, *Mac*, etc);
2. Pode ser escrito em diversos tipos de idiomas (português, inglês, espanhol, alemão, etc);
3. O *VSCode* é totalmente *open-source*, o que garante que todos consigam visualizar, averiguar e editar o código.

Por fim utilizamos o *Doxygen* para gerar a documentação em `HTML` sendo as seguintes algumas das suas características:

1. Pode ser usado em qualquer tipo de sistema operitivo;
2. Suporta vários tipos de linguagens (`C/C++`,`Java`,`Python`,etc);
3. Pode ser criada documentação em formato `HTML`, `PDF`, `UNIX`, entre outros.

3.3 Conclusões

Em suma, o uso dos três programas foi fundamental para a melhor execução do trabalho pois as vantagens de cada um permitiram criar sinergias que trouxeram valor adicional ao resultado obtido.

Capítulo

4

Implementação e Testes

4.1 Introdução

Este capítulo tem o propósito de esclarecer o método utilizado na implementação da solução, para o problema que nos foi apresentado pelo professor. Indica também o método utilizado nos testes realizados, após a implementação estar concluida, na tentaiva de verificar a existência de erros.

4.2 Detalhes de implementação

A implementação deste projeto começou com uma análise ao enunciado do problema que fomos desafiados a solucionar. Após essa análise concluímos que a abordagem inicial deveria passar por implementar um programa no qual os valores fornecidos pelo utilizador fossem armazenados num *array*, para que a sua utilização e leitura se tornasse mais simples ao longo do programa.

4.2.1 Função `main`

A nossa implementação passou por criar uma função `main`, na qual é solicitado ao utilizador que introduza 16 valores, sendo esses valores validados pelo programa. Realiza a verificação dos mesmos (se se encontram no intervalo compreendido entre seis e 14) e apura ainda se estes valores são algarísmos ou letras.

O trecho de código seguinte mostra parte da função `main()`, onde está implementada a funcionalidade descrita anteriormente:

```
do
{
    printf("%d Digite um n mero para o array: ", i);
    int check = scanf("%d", &elemento);
    limpaInput();

    if(check == 1)
    {
        if(elemento < 6 || elemento > 14)
        {
            printf("Numero invalido --> ");
            invalido = 1;
        }
        else
        {
            array[i] = elemento;
            i++;
        }

    }
    else if(check != 1)
    {
        printf("Letras nao sao aceites --> ");
        inputLetra = 1;
    }
}while(i < 16);
```

Excerto de Código 4.1: Trecho de código função `main`

Nesta função está também implementado um menu, no qual o utilizador tem a possibilidade de usufruir das funções presentes no programa criado. Essa implementação foi baseada na utilização de um `switch...case`, onde o programa compara os valores introduzidos pelo utilizador com os casos possíveis, correndo a parte do código correspondente.

4.2.2 Funções utilizadas

As diversas funcionalidades presentes neste programa foram implementadas através de funções, que tornaram possível a sua existência e que ao longo de todo o programa possam ser utilizadas.

A primeira funcionalidade, ***identificação do elemento mais próximo de 15***, está implementada na função `int proximoQuinze`, que compara todos os

valores introduzidos, comparando-os. Apenas necessita de descobrir o maior, pois estes não podem ultrapassar o valor 14, como esse facto é validado na introdução dos valores não é necessário ser feita de novo.

```
for(int i = 0; i < tamVetor; i++){

      if(vetor[i] > max){
          max = vetor[i];
      }
  }
```

Excerto de Código 4.2: Trecho de código função `int proximoQuinze`

A segunda funcionalidade, ***cálculo da multiplicação dos elementos do vetor pelo ultímo elemento***, está integrada a função `int Multiultimoelemento`, sendo que esta multiplica todos os valores introduzidos no vetor pelo ultímo elemento. Esta apenas devolve á função `main` os valores da multiplicação cada posição do vetor, indo buscar os dados à função `main`, onde existe `for loop`, que percorre todos os elemntos do vetor.

A terceira funcionalidade, ou seja, ***criação de uma matriz***, na qual a primeira linha é o vetor original e todas as outras são permutações desta está implementada na função `void matrixPermutacao`. Este trecho do algoritmo começa por imprimir a primeira linha da matriz, referente aos elementos introduzidos pelo utilizador, e depois continua a imprimir as outras linhas, através das permutações feitas na `void baralhaVetor`. Esta função gera números aleatórios para os índices do vetor orginal, imprimindo os valores correspondentes a esses indíces.

A quarta funcionalidade, ***devolução dos valores divisiveis por trs***, foi implementada pela função `int Multiplodetres`, função essa que apenas retorna ao programa o valor da divisão dos valores por três, pois serve para verificar se estes são divisíveis pelo mesmo. Sendo essa verificação feita dentro do `case` correspondente a esta opção, sendo que se o retorno for '0' os números são divisiveis por três, caso contrário não o são.

A quinta funcionalidade, correspondente ao ***cálculo do seno dos elementos do vetor***, está implementado na função `double sind` onde é utilizada uma das funções da biblioteca `math.h`, função esta que calcula o seno dos valores do vetor. Tal como está representado no excerto de código seguinte:

```
double sind (int num){

  double rad, val;

  val = PI / 180;
  rad = sin(num*val);

  return rad;
}
```

Excerto de Código 4.3: Trecho de código função `double sind`

A sexta funcionalidade, ***devolução do vetor introduzido por ordem crescente***, está implementado na função `void ordenaAscendente`, função esta que cria uma cópia do vetor original, tal como está representado no excerto de códico 4.2.2. Fazendo uma comparação entre os valores do vetor original e guardando-os no vetor `copia[i]` por ordem crescente.

```
for (int i = 0; i < tamVetor; i++)
  {
    copia[i] = vetor[i];
  }
```

Excerto de Código 4.4: Trecho de código – cópia vetor original

Foi-nos também proposto a implementação de algumas funcionalidades extra, sendo que estas estão também em diversas funções.

A leitura de um novo vetor e cálculo do seu produto interno é realizada pela função `void vetor`, inicialmente utiliza um processo idntico ao da função `main` para pedir ao utilizador um novo conjunto de vetores e armazená-los num `array`. Realiza, após esta tarefa o produto de todos os valores do vetor, através de uma leitura deste por um ciclo `for`, Imprimindo no ecrã o resultado dessa operação.

A funcionalidade de selecionar os números primos, que tinham sido introduzidos no vetor inicial, é feita através da função `void isprime`, onde é feita a divisão do número introduzido, por um valor 'j', valor este que começa em dois e vai sendo incrementado ao longo da passagem pelos números. Se o resto desta divisão der o valor '0', significa que o número é primo, caso contrário não o é. Esta implementação pode ser vista no excerto de código seguinte:

```
for (int x = 0; x < tamVetor; x++){
    nums[x].num = vetor[x];
    for(int j = 2; j <= nums[x].num/2; j++){

      if(nums[x].num %j == 0){
        resultado++;
      }
    }

    if(resultado == 0){
        nums[x].prime = 1;
    }else{
        nums[x].prime = 0;
    }

    resultado = 0;
    }
```

Excerto de Código 4.5: Trecho de código função `void isprime`

A tarefa de ler um novo vetor e utilizá-lo para devolver ao utilizador uma matriz, na qual os valores apresentados são o produto do vetor inicial com o novo vetor introduzido, valor a valor, está integrada na função `void matrizProduto`, tal como a função `main` e a função `void vetor`, um processo baseado em arrays. Utilizando depois esses valores e ciclos `for` para os percorrer, fazendo o produto de todos entre sí.

4.2.3 Testes

Os testes começaram, primeiramente, por uma revisão ao código escrito numa tentativa de detetar erros de sintaxe que poderiam surgir. O compilador teve também um papel importante nesta parte, pois ajudou a encontrar erros que não tinham sido detetados na leitura do código. As funcionalidades foram testadas pelos elementos do grupo, testanto todos os casos possíveis, mas também os casos limite , tais como a introdução de valores perto dos limites maximos ou mínimos do vetor inicial. Foi também verificado se o programa detetava quando o utilizador inseria valores não numéricos, garantindo que todo o tipo de *inputs* é validado.

4.3 Conclusões

Resumindo, a implementação deste trabalho passou pela criação de um conjunto de `arrays` que foram utilizados para todas as funcionalidades implementadas. A utilização de funções foi fundamental, pelo facto de as diversas tarefas que o programa realiza poderem ser utilizadas em qualquer momento, pelo que esta utilização torna essa tarefa mais simples para os programadores.

Capítulo

5

Conclusões e Trabalho Futuro

5.1 Conclusões Principais

Este trabalho desenvolveu-se a implementação de um programa com novas funcionalidades que facilitam ao utilizador usar para o fim a que se destina de uma forma rápida e eficaz.

Foi atingido o cumprido o objetivo que o grupo de trabalho se propôs, pois permitiu adquirir e aperfeiçoar competências na área de programação que são uma mais valia na engenharia informática.

Portanto, Apesar de cada um dos elementos do grupo ter enriquecido os seus conhecimentos ao realizar o trabalho, todos têm a plena consciência de que é o início de um longo caminho que esperam percorrer e ao longo do qual ainda têm muito a aprender para se tornarem profissionais competentes.

5.2 Trabalho Futuro

Depois de muita troca de ideias entre os elementos do grupo e de muitas tentativas para resolver alguns problemas técnicos, não foi possível terminar o último tópico extra sugerido pelo docente, que se tratava do cálculo do determinante da matriz.

Sem dúvida que um dos próximos objetivos para este trabalho será concluir a última tarefa, ***cálculo do determinante***, introduzir novas funções como a resolução de um sistema através dos seus diferentes métodos de resolução, entre outras, com o objetivo de facilitar os cálculos ao *user*.

Bibliografia

[1] Stephen Wolfram. Wolfram research. *Inc., Mathematica, Version*, 8:23, 2013.

[2] John Hammersley and John Lees-Miller. Overleaf. 2012.

[3] EqsQuest Ltd. Symbolab. 2011.

[4] Matrixcalc.

[5] Tutorialspoint. 2014.

[6] Jeff Atwood and Joel Spolsky. Stackoverflow. 2008.

[1] [2] [3] [4] [5] [6]