



Instituto Politécnico do Cávado e do Ave

Flexible Job Shop

Parte I

Diogo Alexandre Mendes Simões

Projecto de Licenciatura EIM

em Estruturas de Dados Avançadas

Orientador:

Docente João Carlos Silva, IPCA- Instituto Politécnico do Cávado e do Ave

Abril 2022



Flexible Job Shop, C

Diogo Alexandre Mendes Simões

Resumo

Este trabalho, teve como objetivo o desenvolvimento de aplicação em linguagem C, utilizando estruturas de dados dinâmicas.

O trabalho foi dividido em duas partes, que correspondentes a diferentes etapas: Definição de estrutura de um job e operações, Definição de um conjunto finito de jobs e inserção dos mesmos.

Na aplicação desenvolvida na primeira parte, o objetivo é definir um Txt e a realização de operações, tendo que aplicar listas ligadas multidimensionais desenvolvidas em linguagem de programação C.

Palavras-Chave: Estruturas de dados; Linguagem C; Txt; Listas ligadas

Abstract

This work aimed to develop an application in C language, using dynamic data structures.

The work was divided into two parts, corresponding to different stages: Definition of a job's structure and operations, Definition of a finite set of jobs and their insertion.

In the application developed in the first part, the objective is to define a Txt and perform operations, having to apply multidimensional linked lists developed in C programming language.

Keywords: Data structures; C language; txt; linked lists

Índice

Resumo	3
Abstract	4
Introdução.....	7
Objetivos	8
Justificativa / Motivação	8
Contexto	9
Problema.....	10
Descrição da implementação	11
Bibliotecas	11
Descrição Técnica.....	11
Desenvolvimento	13
Armazenamento do ficheiro de dados.....	14
Funções	15
Inicialização do Programa	15
Inserir uma nova Operação.....	16
Visualização de Jobs	17
Remover uma determinada operação	19
Alteração de uma determinada operação	20
Determinar a quantidade mínima de unidades de tempo necessárias para completar o job e listagem das respetivas operações.....	23
Determinação da quantidade máxima de unidades de tempo necessárias para completar o job e listagem das respetivas operações;	23
Determinação da quantidade média de unidades de tempo necessárias para completar uma operação, considerando todas as alternativas possíveis;	24
Conclusão	25
Bibliografia.....	26

Índice de Figuras

Figura 1-Tabela FJSSP	10
Figura 2-Estruturas	12
Figura 3-Diagrama Simplificação	13
Figura 4-Ficheiro Txt.....	14
Figura 5-Menu Principal.....	15
Figura 6-Adicionar Operação.....	16
Figura 7-Adicionar Máquina.....	16
Figura 8-Máquina(Id, tempo).....	17
Figura 9-Main Menu.....	17
Figura 10-Visualização Operações.....	18
Figura 11-Remover Operação	19
Figura 12-Confirmação de Remoção.....	19
Figura 13-Confirmação Remover.....	20
Figura 14-Alterar Operação	20
Figura 15-Menu de Alteração	21
Figura 16-Modificar Operação.....	21
Figura 17-Criação de uma Máquina na Operação.....	22
Figura 18-Troca de um novo tempo em uma máquina.....	22
Figura 19-Remover uma máquina.....	22
Figura 21-Lista Operação Base Txt.....	22
Figura 20-Lista Operação Alterada	22
Figura 22-Tempo Mínimo Job	23
Figura 23-Tempo Máximo cada Job.....	23
Figura 24-Tempo Máximo cada Operação	24

Introdução

No âmbito da unidade curricular de Estruturas de dados avançadas, do segundo semestre do primeiro ano do curso de Engenharia Informática Médica do Instituto Politécnico do Cavado e do Ave, no ano letivo 2021/2022 foi-nos proposto a elaboração de um trabalho prático que está dividido em duas partes.

Este trabalho surge no contexto da avaliação contínua composta por um trabalho prático. O objetivo principal deste trabalho é levar ao melhor entendimento dos conhecimentos adquiridos ao longo do semestre e a melhor aplicação dos mesmos.

Assim, com este trabalho pretendemos sedimentar mais os nossos conhecimentos sobre os temas abordados nas respetivas aulas.

O trabalho realizado individual tem como objetivo a elaboração de um programa em C para o problema de escalonamento denominado Flexible Job Shop Problem (FJSSP).

Objetivos

Espera-se, com a execução desse projeto, sedimentar os conhecimentos relativos à definição e manipulação de estruturas de dados dinâmicos em linguagem de programação C no desenvolvimento de uma solução digital para o problema de escalonamento denominado por Flexible Job Shop Problem, permitindo gerar uma proposta de escalonamento para a produção de um produto envolvendo várias operações e a utilização de várias máquinas.

Justificativa / Motivação

A intensa disputa por novos mercados e o apogeu das novas tecnologias tem obrigado as empresas a desenvolverem constantes estratégias que possam dinamizar seus processos internos e inovar os seus sistemas.

O Software a ser desenvolvida visa prestar suporte as organizações focadas no processo são oficinas de artesanato, restaurantes, empresas de mecânica, oficinas de pintura, gráficas comerciais e outras indústrias que fabricam produtos personalizados em pequenos lotes.

Contexto

Neste projeto que nos foi proposto, tendo origem num trabalho prático a avaliar na disciplina do 1º ano do curso LEIM, Estruturas de dados avançadas, que está dividido em duas partes, iremos abordar neste relatório apenas a primeira parte no projeto.

Nesta parte do projeto será necessário realizar os seguintes requisitos:

- Definição de uma estrutura de dados dinâmica para a representação de um job com um conjunto finito de n operações;
- Armazenamento/leitura de ficheiro de texto com representação de um job;
- Inserção de uma nova operação;
- Remoção de uma determinada operação;
- Alteração de uma determinada operação;
- Determinação da quantidade mínima de unidades de tempo necessárias para completar o job e listagem das respetivas operações;
- Determinação da quantidade máxima de unidades de tempo necessárias para completar o job e listagem das respetivas operações;
- Determinação da quantidade média de unidades de tempo necessárias para completar uma operação, considerando todas as alternativas possíveis;

Problema

Foi fornecido um projeto com um problema do tipo FJSSP, que este por sua vez representa tarefas de planos de processos variáveis, também conhecidos como Empregos.

Cada plano de processo, possui um conjunto de operações que devem ser feitas por ordem, não tendo a opção de saltar operações.

Cada máquina pode realizar apenas 1 operação por vez.

Neste projeto, falar-se-á apenas da primeira parte, por ser a implementada.

Process Plan	Operation						
	0 1	0 2	0 3	0 4	0 5	0 6	0 7
pr _{1,2}	(1,3) [4,5]	(2,4) [4,5]	(3,5) [5,6]	(4,5,6,7,8) [5,5,4,5,9]			
pr _{2,2}	(1,3,5) [1,5,7]	(4,8) [5,4]	(4,6) [1,6]	(4,7,8) [4,4,7]	(4,6) [1,2]	(1,6,8) [5,6,4]	(4) [4]
pr _{3,3}	(2,3,8) [7,6,8]	(4,8) [7,7]	(3,5,7) [7,8,7]	(4,6) [7,8]	(1,2) [1,4]		
pr _{4,2}	(1,3,5) [4,3,7]	(2,8) [4,4]	(3,4,6,7) [4,5,6,7]	(5,6,8) [3,5,5]			
pr _{5,1}	(1) [3]	(2,4) [4,5]	(3,8) [4,4]	(5,6,8) [3,3,3]	(4,6) [5,4]		
pr _{6,1}	(1,2,3)	(4,5)	(3,6)				

Figura 1-Tabela FJSSP

Descrição da implementação

Para desenvolver o programa e para que este funcione corretamente, foi necessário que todos os procedimentos e funções criados interagissem de forma eficaz, sem erros e evitando situações do tipo “beco sem saída” que ocorrem quando as aplicações bloqueiam ou ficam presas em estados sem retorno. O resultado final é o (código) fonte que se encontra anexado a este relatório, sendo que nesta sessão serão explicadas algumas técnicas implementadas.

Bibliotecas

```
#include <stdlib.h>
#include <stdio.h>
#include <stdbool.h>
#include <locale.h>
#include <ctype.h>
#include <string.h>
```

Estas instruções permitem incluir bibliotecas ao programa. Além das necessidades normais de input/output ([stdio.h](#)), alocação de memória ([stdlib.h](#)), manipulação de strings ([string.h](#)) e operações com diferentes tipos de variáveis ([ctype.h](#)), foi necessário recorrer à biblioteca [locale.h](#) para poder incluir caracteres especiais em língua portuguesa e à biblioteca ([stdbool.h](#)) recurso true e false, que serão largamente utilizados ao longo do programa.

Descrição Técnica

Para a realização deste trabalho foi necessário recorrer à criação de diversas estruturas.

Uma estrutura é um conjunto de variáveis de tipos distintos ou não, agrupadas sob um único nome. As variáveis que compõem a estrutura são chamadas membros, campos ou elementos.

Com a palavra-chave **struct** declara-se um novo tipo de dado.

A palavra seguinte será o seu identificador ou tipo de estrutura.

Quando se faz a declaração **struct data**:

- **struct** é um tipo distinto de dado como int ou float.
- **data** é referência a um tipo específico de estrutura.

Os membros são declarados entre chaves. A declaração termina com um “ponto e vírgula” (;).

Foram criadas as seguintes estruturas:

- Job
- Máquina
- Operação

```
//Jobs
typedef struct Job
{
    int Id; //Contém um ID
    struct Operacao* Operacoes; //Contém NºOperações
    struct Job* Next; //Próximo Job
} Job;
//Máquina
typedef struct Máquina
{
    int Id; //Contém ID
    int Tempo; //Contém Tempo
    struct Máquina* Next; //Próxima Máquina
} Máquina;
//Operação
typedef struct Operacao
{
    int Id; //Contém um ID
    struct Máquina* Maquinas; //Contém NºMaquinas
    struct Operacao* Next; // Próxima Operação
} Operacao;
```

Figura 2-Estruturas

Desenvolvimento

Para a inicialização de este projeto, o primeiro a ser feito seria entender o problema em si e o que solicitava, sendo uma das partes mais complexas e demoradas.

Determinamos que um determinado Job tem varias operações que estas mesmas tem vários processos, nomeadamente dividido em Tempo e a Maquina que o executa.

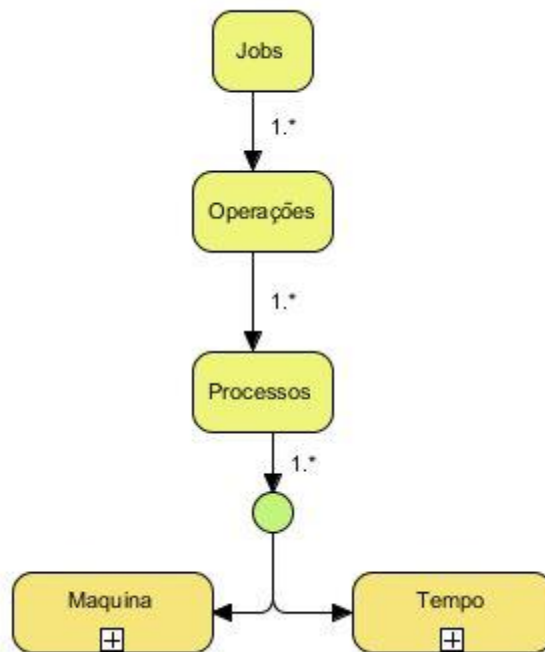
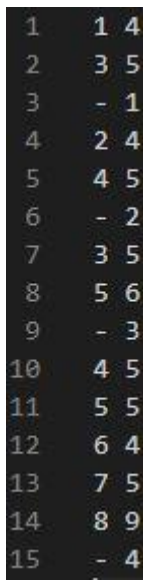


Figura 3-Diagrama Simplificação

Armazenamento do ficheiro de dados

O armazenamento de dados em ficheiro Txt, que a ideia principal seria implementar em ficheiro Json, o formato do registo é Maquina -> Tempo.



1	1	4
2	3	5
3	-	1
4	2	4
5	4	5
6	-	2
7	3	5
8	5	6
9	-	3
10	4	5
11	5	5
12	6	4
13	7	5
14	8	9
15	-	4

Figura 4-Ficheiro Txt

Ao processamento do arquivo, utilizamos especificadores do tipo fscanf, %d, este é o espaço reservado mais usado na língua C. É usado para ler o valor inteiro.

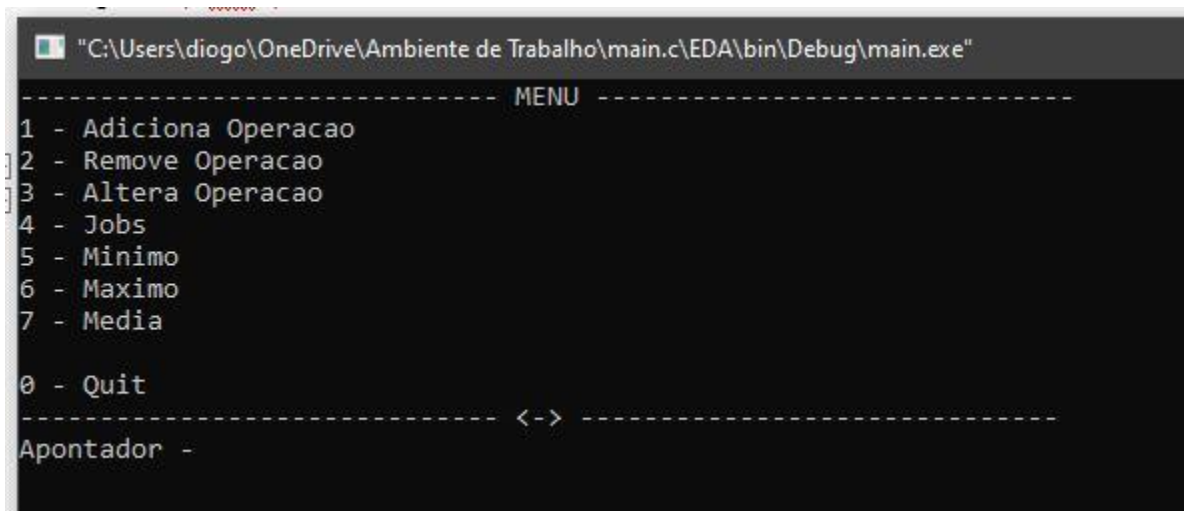
- A função scanf() é usada para ler entrada formatada de stdin na linguagem C. Ele retorna todo o número de caracteres escritos nele de outra forma, retorna um valor negativo.
- A função fscanf() é usada para ler a entrada formatada a partir do fluxo dado na língua C. Ele retorna zero, se não tiver sucesso. Caso contrário, ele retorna a sequência de entrada, se for bem-sucedido.

Funções

Na linguagem C, a função é um conjunto de comandos que realiza uma tarefa específica num módulo dependente de código. A função é referenciada pelo programa principal através do nome atribuído a ela. A utilização de funções visa modelizar um programa em várias partes, no qual é muito comum em programação estruturada. Desta forma podemos dividir um programa em várias partes, no qual cada função realiza uma tarefa bem definida. No nosso programa está implementadas as seguintes funções:

Inicialização do Programa

Para ajudar o utilizador, realizou-se um “Main-Menu”, para que o utilizador possa navegar de uma maneira simplificada para poder manipular os dados.



```
"C:\Users\diogo\OneDrive\Ambiente de Trabalho\main.c\EDA\bin\Debug\main.exe"
----- MENU -----
1 - Adiciona Operacao
2 - Remove Operacao
3 - Altera Operacao
4 - Jobs
5 - Minimo
6 - Maximo
7 - Media
0 - Quit
----- <-> -----
Apontador -
```

Figura 5-Menu Principal

Inserir uma nova Operação

Selecionando a opção “Adicionar Operação” o utilizador poderá escolher o Id que irá atribuir à operação.

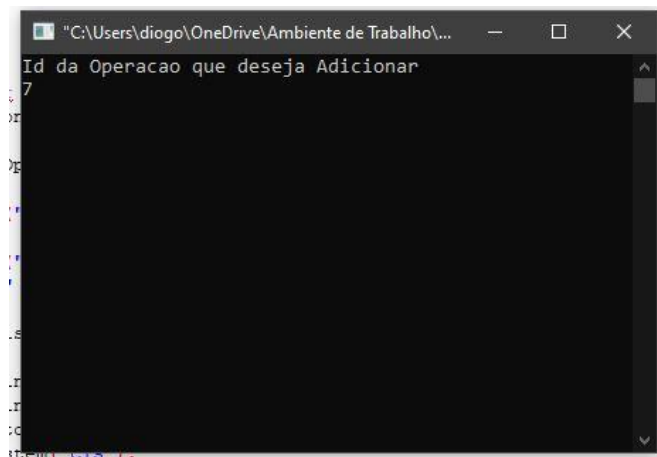


Figura 6-Adicionar Operação

Uma vez introduzido o Id que queremos dar à operação, teremos a possibilidade de adicionar uma máquina à operação.

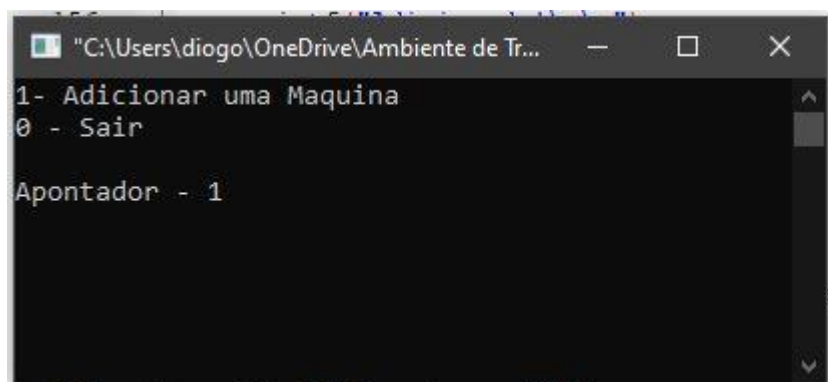


Figura 7-Adicionar Maquina

Introduzindo o Id da máquina, ele possibilitará a introdução do tempo que queremos de esta determinada máquina.

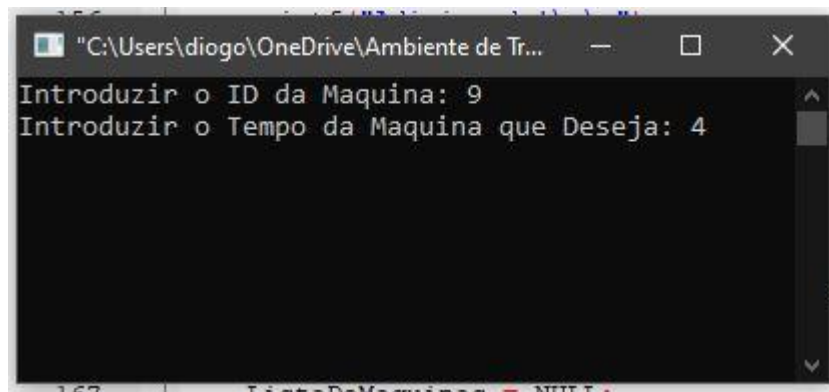


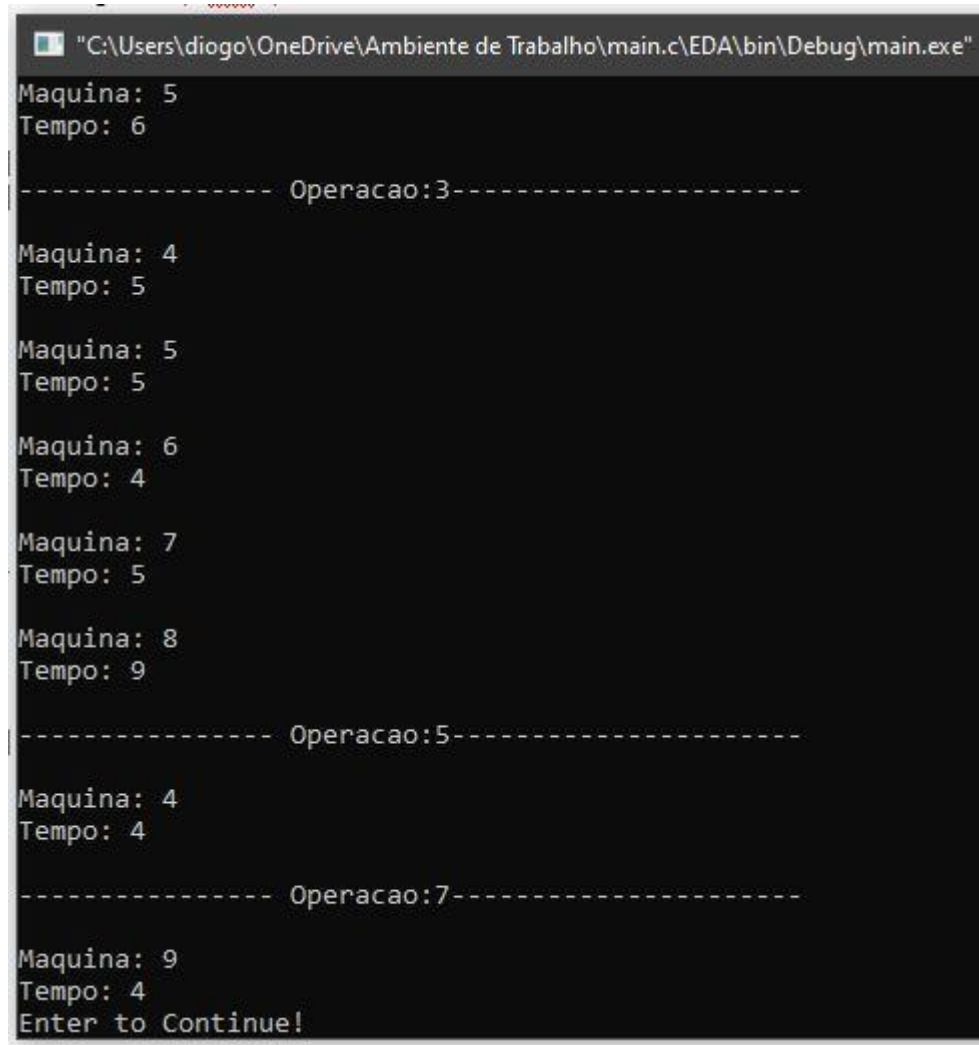
Figura 8-Maquina(Id, tempo)

Visualização de Jobs



Figura 9-Main Menu

O utilizador poderá nesta área poder visualizar as várias operações associadas ao único Job existente.



```
"C:\Users\diogo\OneDrive\Ambiente de Trabalho\main.c\EDA\bin\Debug\main.exe"
Maquina: 5
Tempo: 6

----- Operacao:3-----

Maquina: 4
Tempo: 5

Maquina: 5
Tempo: 5

Maquina: 6
Tempo: 4

Maquina: 7
Tempo: 5

Maquina: 8
Tempo: 9

----- Operacao:5-----

Maquina: 4
Tempo: 4

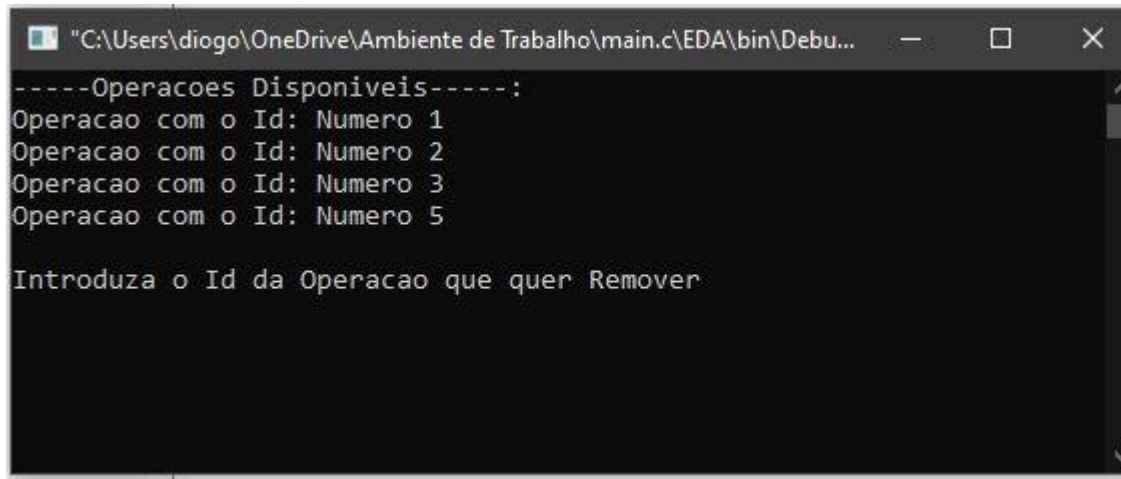
----- Operacao:7-----

Maquina: 9
Tempo: 4
Enter to Continue!
```

Figura 10-Visualização Operações

Remover uma determinada operação

Para remover uma determinada operação, basta introduzir o Id que lhe corresponde.

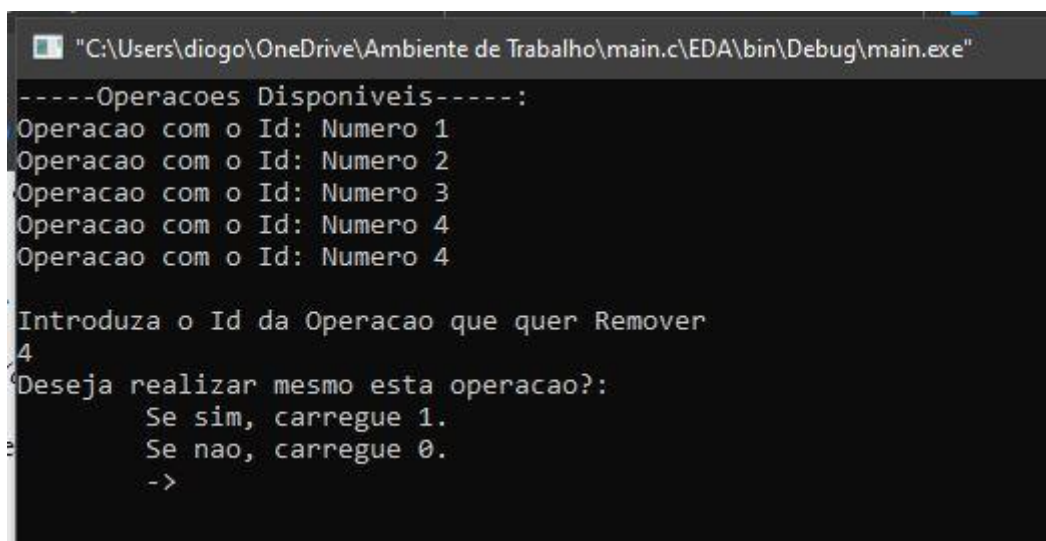


```
"C:\Users\diogo\OneDrive\Ambiente de Trabalho\main.c\EDA\bin\Debu...
-----Operacoes Disponiveis-----:
Operacao com o Id: Numero 1
Operacao com o Id: Numero 2
Operacao com o Id: Numero 3
Operacao com o Id: Numero 5

Introduza o Id da Operacao que quer Remover
```

Figura 11-Remover Operação

O sistema para verificação a decisão, solicitará uma confirmação para remover a operação em questão.



```
"C:\Users\diogo\OneDrive\Ambiente de Trabalho\main.c\EDA\bin\Debug\main.exe"
-----Operacoes Disponiveis-----:
Operacao com o Id: Numero 1
Operacao com o Id: Numero 2
Operacao com o Id: Numero 3
Operacao com o Id: Numero 4
Operacao com o Id: Numero 4

Introduza o Id da Operacao que quer Remover
4
Deseja realizar mesmo esta operacao?:
    Se sim, carregue 1.
    Se nao, carregue 0.
->
```

Figura 12-Confirmação de Remoção

Uma vez confirmada a decisão de apagar, o sistema o realizará e enviará uma mensagem de confirmação.

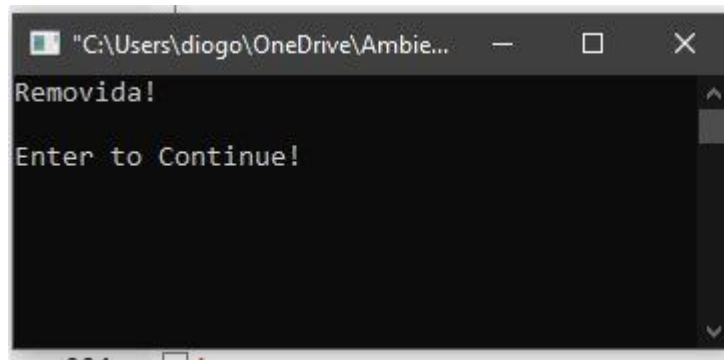


Figura 13-Confirmação Remove

Alteração de uma determinada operação

O utilizador terá acesso à área de alteração de uma determinada operação, para ajudar o utilizador, será apresentado a lista de operações existentes.

Caso queira realizar uma operação, basta inserir o Id da respetiva Operação que deseja modificar.

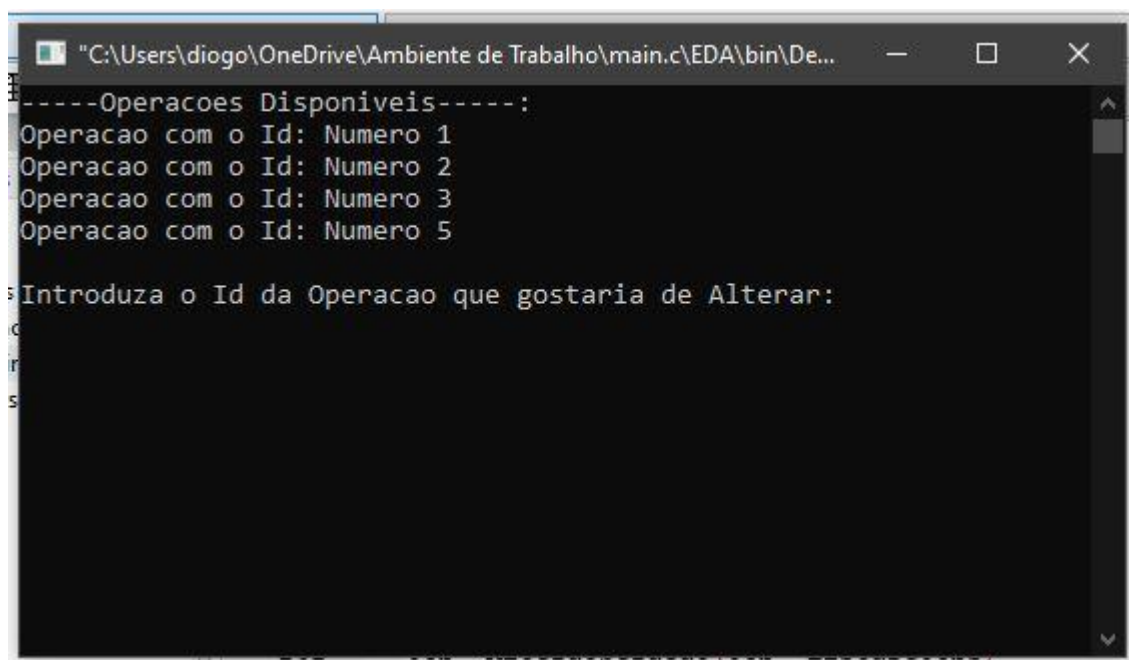


Figura 14-Alterar Operação

Caso de sucesso, será redirecionado para um novo menu, onde poderá editar uma operação.

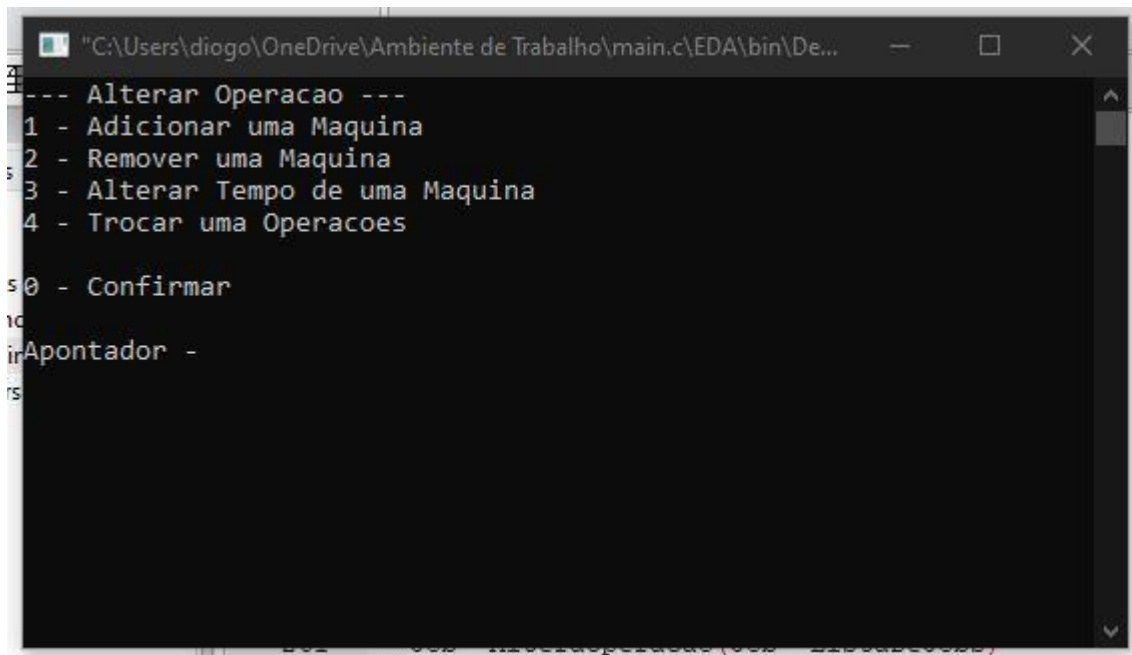


Figura 15-Menu de Alteração

Caso o utilizador selecione a opção 1, será proposto a troca de Id e o tempo da máquina.

Se o utilizador seleccionar as demais opções, será redirecionado para a alteração das mesmas.

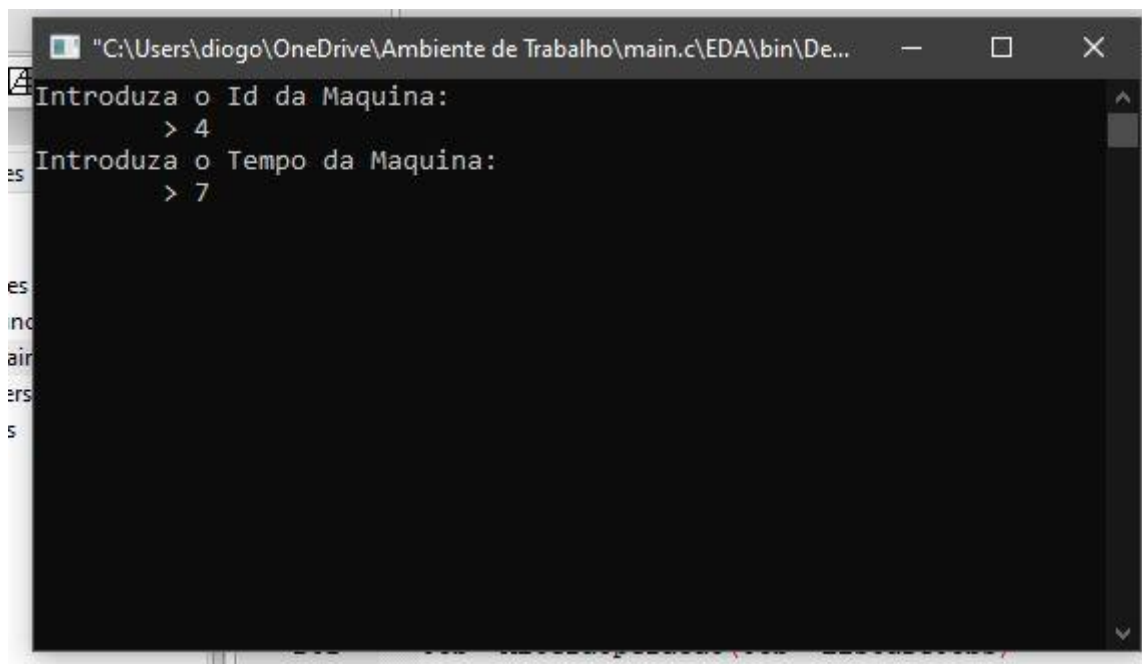


Figura 16-Modificar Operação

Na seguinte sequencia iremos abordar todas as operações de uma alteração numa operação. Primeiramente iremos criar uma nova maquina com o Id 9 e o tempo de 10.

```
"C:\Users\diogo\OneDrive\Ambiente de Trabalho\main.c\ED
Introduza o Id da Maquina:
> 9
Introduza o Tempo da Maquina:
> 10
```

Figura 17-Criação de uma Máquina na Operação

Seguidamente iremos trocar o tempo de uma máquina.

```
"C:\Users\diogo\OneDrive\Ambiente de Trabalho\main.c\ED
Introduza o Id da Maquina:
> 5
Insira o novo Tempo da Maquina Escolhida:
> 1
Tempo MudadoEnter to Continue!
```

Figura 18-Troca de um novo tempo em uma maquina

De seguida iremos remover uma determinada maquina.

```
"C:\Users\diogo\OneDrive\Ambiente
Introduza o Id da Maquina:
> 4
```

Figura 19-Remover uma maquina

```
----- Operac
Maquina: 4
Tempo: 5

Maquina: 5
Tempo: 5

Maquina: 6
Tempo: 4

Maquina: 7
Tempo: 5

Maquina: 8
Tempo: 9

----- Operac
```

Figura 20-Lista Operação Base Txt

->

```
----- Operac
Maquina: 5
Tempo: 1

Maquina: 6
Tempo: 4

Maquina: 7
Tempo: 5

Maquina: 8
Tempo: 9

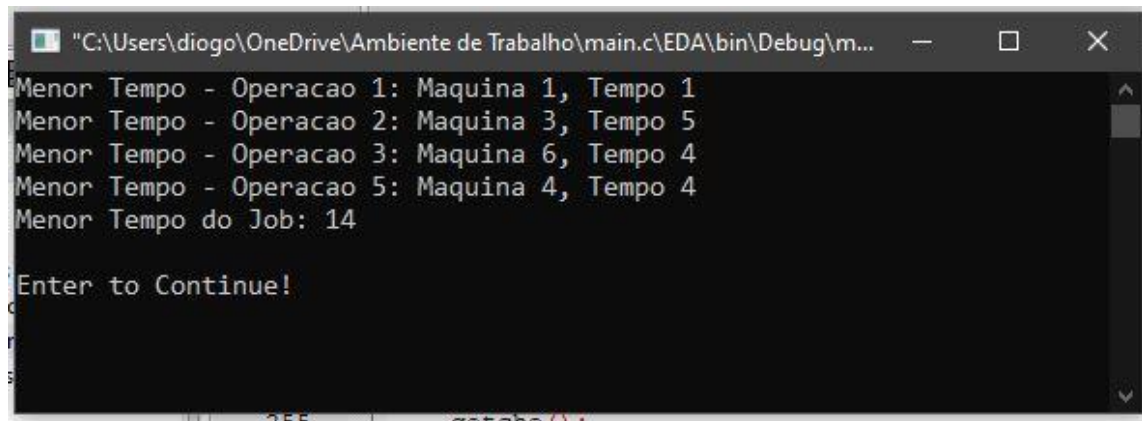
Maquina: 9
Tempo: 10

----- Operac
```

Figura 21-Lista Operação Alterada

Determinar a quantidade mínima de unidades de tempo necessárias para completar o job e listagem das respetivas operações

O utilizador terá acesso à área de visualizar o menor tempo para completar um determinado Job, para ajudar o utilizador, será apresentado a lista de operações existentes.



```

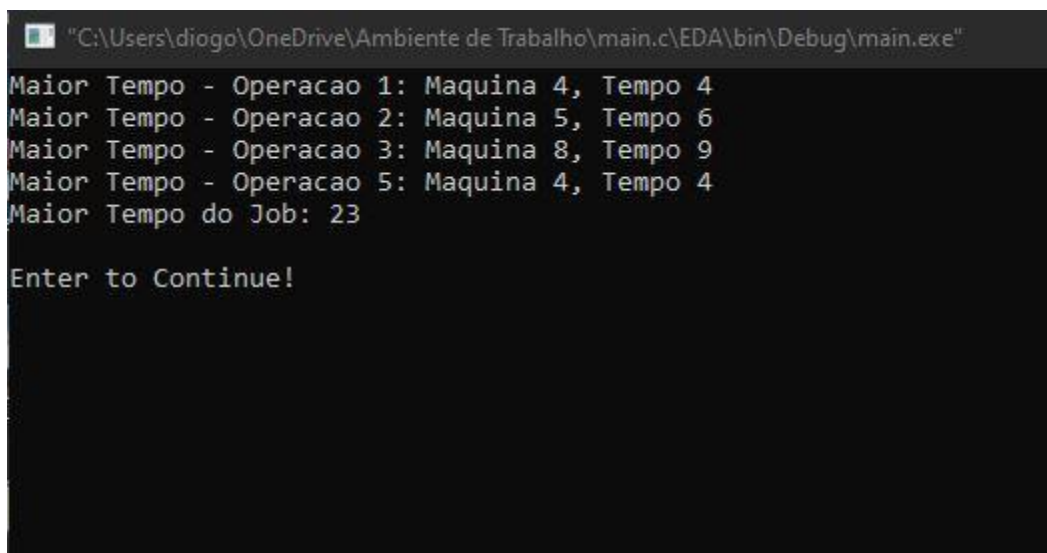
"C:\Users\diogo\OneDrive\Ambiente de Trabalho\main.c\EDA\bin\Debug\main.exe"
Menor Tempo - Operacao 1: Maquina 1, Tempo 1
Menor Tempo - Operacao 2: Maquina 3, Tempo 5
Menor Tempo - Operacao 3: Maquina 6, Tempo 4
Menor Tempo - Operacao 5: Maquina 4, Tempo 4
Menor Tempo do Job: 14

Enter to Continue!
  
```

Figura 22-Tempo Minimo Job

Determinação da quantidade máxima de unidades de tempo necessárias para completar o job e listagem das respetivas operações;

O utilizador terá acesso à área de visualizar o maior tempo para completar um determinado Job, para ajudar o utilizador, será apresentado a lista de operações existentes.



```

"C:\Users\diogo\OneDrive\Ambiente de Trabalho\main.c\EDA\bin\Debug\main.exe"
Maior Tempo - Operacao 1: Maquina 4, Tempo 4
Maior Tempo - Operacao 2: Maquina 5, Tempo 6
Maior Tempo - Operacao 3: Maquina 8, Tempo 9
Maior Tempo - Operacao 5: Maquina 4, Tempo 4
Maior Tempo do Job: 23

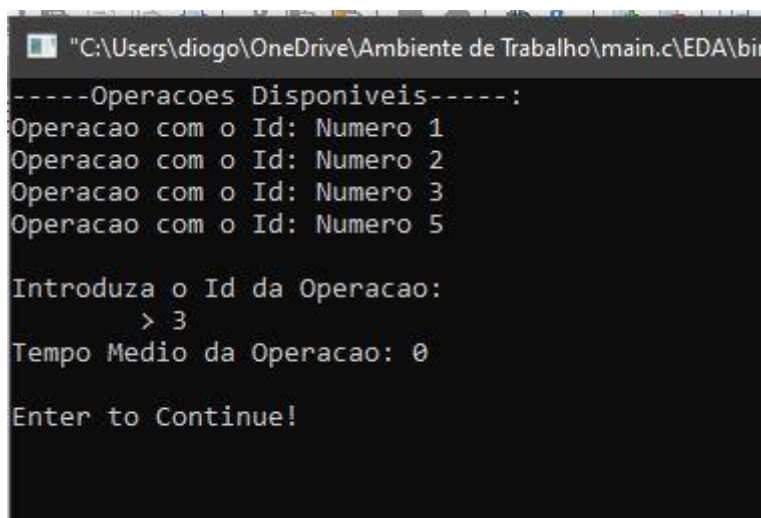
Enter to Continue!
  
```

Figura 23-Tempo Máximo cada Job

Determinação da quantidade média de unidades de tempo necessárias para completar uma operação, considerando todas as alternativas possíveis;

O utilizador terá acesso à área de consultar a média de uma determinada operação, para ajudar o utilizador, será apresentado a lista de operações existentes.

Caso queira realizar uma operação, basta inserir o Id da respetiva Operação que deseja consultar.



```
"C:\Users\diogo\OneDrive\Ambiente de Trabalho\main.c\EDA\bin
-----Operacoes Disponiveis-----:
Operacao com o Id: Numero 1
Operacao com o Id: Numero 2
Operacao com o Id: Numero 3
Operacao com o Id: Numero 5

Introduza o Id da Operacao:
    > 3
Tempo Medio da Operacao: 0

Enter to Continue!
```

Figura 24-Tempo Máximo cada Operação

A média será = à Soma / Todos os elementos presentes em cada operação.

Nesta parte, o resultado final não foi o esperado, tendo como valor médio 0 para todas as operações seleccionadas.

Este problema ainda não foi corrigido devido a não conseguir detetar a falha para poder ter esse resultado, no qual ainda trabalharei para poder corrigir futuramente.

Conclusão

Após terminarmos a realização do trabalho, conseguimos pôr em prática os conhecimentos dados nas aulas, embora nos tenhamos deparado com bastantes dificuldades.

Apesar de todas as dificuldades encontradas neste trabalho, conseguimos superar algumas com sucesso.

Este trabalho consegue-nos demonstrar um pouco o nível de dificuldade que podemos ter no exterior a nível profissional.

Contudo, este trabalho vem nos trazer mais experiência e conhecimento da disciplina, preparando-nos assim para a realização de mais trabalhos no futuro com outros programadores/empresas.

Para consultar a documentação e código, será disponibilizado este link GitHub para poder acompanhar o processo e os feitos do mesmo.

Link : <https://github.com/DiogoAlex960/EDA>

Bibliografia

Na realização deste trabalho, fomos encontrando várias dificuldades ao longo do projeto, e quase todos ultrapassados com sucesso, como foi dito anteriormente.

Usamos assim livros de programação, os conteúdos e códigos usados nas aulas da Unidade Curricular Estrutura de dados Avançadas.

O que é FJSP - <https://www.localsolver.com/docs/last/exampletour/flexiblejobshop.html>

<https://www.youtube.com/watch?v=SNE36xuGfV8>

<https://www.youtube.com/watch?v=B36tjfUJNBM>

Bibliotecas C - <https://www.ime.usp.br/~pf/algoritmos/apend/interfaces.html>

https://www.tutorialspoint.com/c_standard_library/index.htm

Corrigir Código C - <https://docs.microsoft.com/pt-br/cpp/code-quality/walkthrough-analyzing-c-cpp-code-for-defects?view=msvc-170>

Duvidas sobre Funções - [Stack Overflow - Where Developers Learn, Share, & Build Careers](#)

DoxyGen - <https://www.youtube.com/watch?v=R150qI6e7HU>

