

**Relatório Fase 2**

**Estrutura de Dados Avançadas**

**Alunos:**

**18450 – Rui Pinto**

**Professor: João Carlos Silva;**

**Licenciatura em Engenharia de Sistemas Informaticos**

Barcelos, Junho, 2022

Índice

[Introdução 3](#_Toc105014995)

[Propósitos e objetivos 4](#_Toc105014996)

[Estrutura de Dados 5](#_Toc105014997)

[Main.c 5](#_Toc105014998)

[Header.h 7](#_Toc105014999)

[Functions.c 9](#_Toc105015000)

[Iniciar Operação / Iniciar Job 9](#_Toc105015001)

[Listar Jobs 10](#_Toc105015002)

[Inserir Job 11](#_Toc105015003)

[Remover Job 14](#_Toc105015004)

[Inserir operação num determinado Job 15](#_Toc105015005)

[Remover operação de um determinado Job 16](#_Toc105015006)

[Editar Operação de um determinado Job 17](#_Toc105015007)

[Testes realizados 18](#_Toc105015008)

[Conclusão 22](#_Toc105015009)

[Bibliografia 22](#_Toc105015010)

Índice das Imagens

[Figura 1 - Bibliotecas e ficheiros Main.c 5](#_Toc105015011)

[Figura 2 - Variáveis e inicio do main 5](#_Toc105015012)

[Figura 3 - Opções do utilizador e fim do main 6](file:///C:\Users\ASUS\Desktop\RelatorioEDA.docx#_Toc105015013)

[Figura 4 - Bibliotecas Header.h 7](#_Toc105015014)

[Figura 5 - Structs 7](#_Toc105015015)

[Figura 6 - Assinaturas 8](file:///C:\Users\ASUS\Desktop\RelatorioEDA.docx#_Toc105015016)

[Figura 7 - InicializarOperação 9](#_Toc105015017)

[Figura 8 - InicializarJob 9](#_Toc105015018)

[Figura 9 - ListarJobEmListaOrdenada 10](#_Toc105015019)

[Figura 10 - ListarNoJob 10](#_Toc105015020)

[Figura 11 - ListarNoOperação 11](#_Toc105015021)

[Figura 12 - InserirJob 11](#_Toc105015022)

[Figura 13 - NovoNoJob 12](#_Toc105015023)

[Figura 14 - NovoNoOperacao 12](#_Toc105015024)

[Figura 15 - InsereNoOperacao 13](#_Toc105015025)

[Figura 16 - LeOperacao 13](#_Toc105015026)

[Figura 17 - RemoveJob 14](#_Toc105015027)

[Figura 18 - DescobreJob 15](#_Toc105015028)

[Figura 19 - RemoveOperacao 16](file:///C:\Users\ASUS\Desktop\RelatorioEDA.docx#_Toc105015029)

[Figura 20 - EditaOperacao 17](#_Toc105015030)

[Figura 21 - Menu 18](#_Toc105015031)

[Figura 22 - Adicionar operações ao job 18](#_Toc105015032)

[Figura 23 - Maquinas utilizadas na operação 19](#_Toc105015033)

[Figura 24 - Dados da operação 19](#_Toc105015034)

[Figura 25 - Job adicionado 19](#_Toc105015035)

[Figura 26 - Dados para atualizar 19](#_Toc105015036)

[Figura 27 - Atualização de dados 20](#_Toc105015037)

[Figura 28 - Operação editada 20](#_Toc105015038)

[Figura 29 - Dados atualizados 20](#_Toc105015039)

[Figura 30 - Remoção de uma operação 21](#_Toc105015040)

[Figura 31 - Listagem atualizada 21](#_Toc105015041)

[Figura 32 - Listagem final 21](#_Toc105015042)

# Introdução

O trabalho desenvolvido no âmbito da unidade curricular “Estrutura de Dados Avançadas”, tem como objetivo desenvolvimento de uma solução digital para o problema de escalonamento denominado Flexible Job Shop Problem. O programa permite gerar uma proposta de escalonamento para a produção de um produto envolvendo várias operações e a utilização de várias máquinas, minimizando o tempo as unidades de tempo necessário na sua produção.

Este projeto tem ainda a finalidade de aplicar conteúdos lecionados anteriormente da linguagem C, aprofundando-os, e ainda aplicar novos conteúdos como algoritmos de procura e de ordenação, listas dinâmicas e apontadores.

# Propósitos e objetivos

Tal como foi referido, o objetivo do programa é gerir a produção de um determinado produto que é tratado como um Job, com as várias operações necessárias e as máquinas que serão utilizadas.

Na fase 1 do projeto foi-nos proposto gerir a produção de apenas um produto, podendo adicionar, editar ou remover as operações e determinar o tempo máximo, mínimo e médio necessário para completar o Job, apresentando as respetivas operações.

O programa também terá de abrir o ficheiro de texto que terá guardado toda a informação necessária para depois caso seja preciso modificar alguma informação ou até mesmo adicionar operações. No fim a informação será guardada no mesmo ficheiro.

A fase 2 do projeto gere a produção de vários produtos, tendo como objetivo adicionar, editar ou remover os Jobs. No fim do programa, toda a nova informação será armazenada/atualizada no documento de texto dados.txt.

# Estrutura de Dados

A estrutura do projeto é composta por 3 ficheiros (Main.c , Header.h , Functions.c) todos realizados com a linguagem C.

## Main.c

Este é o programa principal porque é onde corre o programa.

Primeiramente é constituído pelas bibliotecas necessárias para o funcionamento do programa e os restantes ficheiros do projeto (“Header.h” e “functions.c”).

Uma imagem com texto

Descrição gerada automaticamente

Figura 1 - Bibliotecas e ficheiros Main.c

Depois de incluir as bibliotecas e os restantes ficheiros começa a função main (a função que principal). No início da função contém todas as variáveis necessárias e a utilização de funções para criar uma lista para os jobs e as operações e abrir o ficheiro de texto para recolher os dados.

Uma imagem com texto

Descrição gerada automaticamente

Figura 2 - Variáveis e inicio do main

Por fim irá aparecer o menu e depois da recolha da opção do utilizador irá efetuar a função pretendida. No fim do programa os dados serão todos atualizados.

Uma imagem com texto

Descrição gerada automaticamenteUma imagem com texto

Descrição gerada automaticamente

Figura 3 - Opções do utilizador e fim do main

## Header.h

O ficheiro Header.h é utilizado como biblioteca para guardar as structs e as assinaturas das funções utilizadas durante todo o projeto. Tal como no main.c, no início terá de se colocar todas as bibliotecas necessárias para a execução do projeto.

Uma imagem com texto

Descrição gerada automaticamente

Figura 4 - Bibliotecas Header.h

De seguida, estão as struct utilizadas no projeto (uma para a representação dos jobs e outra para as operações).

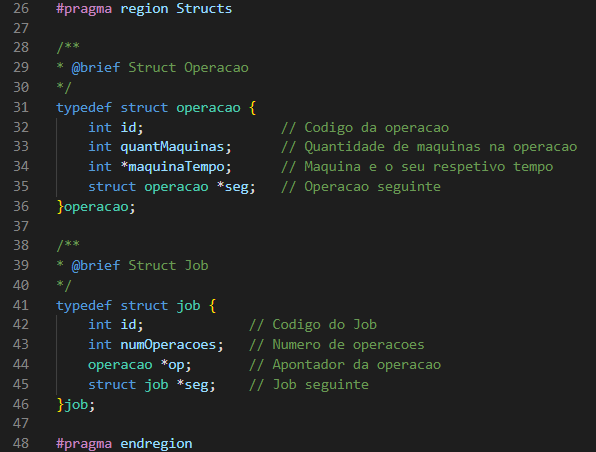


Figura 5 - Structs

Para finalizar estão colocadas todas as funções utilizadas no projeto divididas por categorias (Operações, Jobs, Ficheiros e outras funções).

Uma imagem com texto

Descrição gerada automaticamente

Uma imagem com texto, captura de ecrã, ecrã

Descrição gerada automaticamente

Figura 6 - Assinaturas

## Functions.c

No último ficheiro a apresentar trata-se do functions.c, que não é nada mais nada menos que o ficheiro utilizado para guardar todas as funções existentes neste projeto

### Iniciar Operação / Iniciar Job

Estas funções são utilizadas para abrir/criar uma lista para as operações e os Jobs.

Uma imagem com texto

Descrição gerada automaticamente

Figura 7 - InicializarOperação

Uma imagem com texto

Descrição gerada automaticamente

Figura 8 - InicializarJob

### Listar Jobs

Para listar os Jobs existentes são utilizadas as seguintes funções:

* **ListarJobEmListaOrdenada:** Esta função é puxada no main e percorre Job a Job para listar.

Uma imagem com texto

Descrição gerada automaticamente

Figura 9 - ListarJobEmListaOrdenada

* **ListarNoJob:** A função seguinte conta as operações existentes do job.

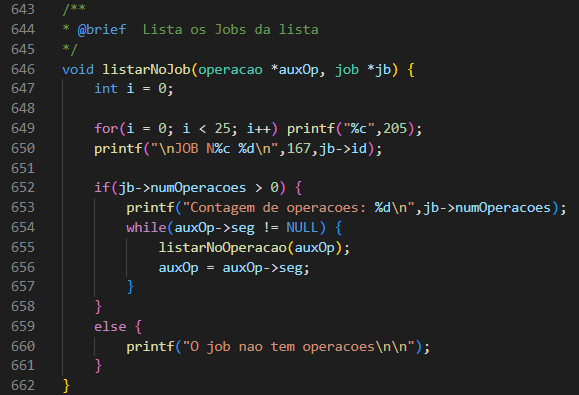


Figura 10 - ListarNoJob

* **ListarNoOperação:** A seguinte função lista todas as operações do Job selecionado.

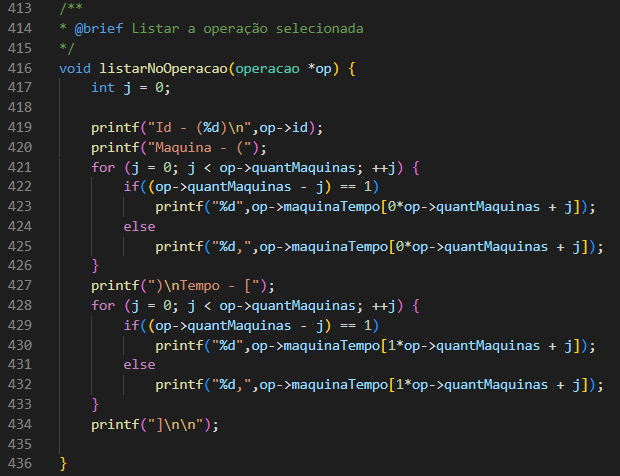


Figura 11 - ListarNoOperação

### Inserir Job

Para inserir um Job são utilizadas as seguintes funções:

* **InserirJob:** A seguinte função começa com incrementar mais um Job e depois de o criar questiona o utilizador se pretende já adicionar as operações aos Jobs.

Uma imagem com texto

Descrição gerada automaticamente

Figura 12 - InserirJob

* **NovoNoJob:** Esta função cria o Job na lista.

Uma imagem com texto

Descrição gerada automaticamente

Figura 13 - NovoNoJob

* **NovoNoOperacao:** Tal como foi utilizado para criar um job na lista, também será necessário criar uma operação na lista, e é essa a funcionalidade da função seguinte.

Uma imagem com texto

Descrição gerada automaticamente

Figura 14 - NovoNoOperacao

* **InsereNoOperacao:** Caso o utilizador pretenda adicionar as operações, a seguinte função criará operações ao job.

Uma imagem com texto

Descrição gerada automaticamente

Figura 15 - InsereNoOperacao

* **LeOperacao:** Por último esta função irá perguntar ao utilizador as informações acerca das operações.

Uma imagem com texto

Descrição gerada automaticamente

Figura 16 - LeOperacao

### Remover Job

Para remover um Job é utilizada a seguinte função:

* **RemoveJob:** A função irá pedir primeiramente o código do Job que o utilizador pretende remover, e de seguida irá mostra os dados do Job que foi removido. Caso dê um erro a função avisa que o código é inexistente.

Uma imagem com texto

Descrição gerada automaticamente

Figura 17 - RemoveJob

### Inserir operação num determinado Job

Para inserir uma operação num determinado job são utilizadas as seguintes funções:

**DescobreJob:** Esta função é utilizada para pedir o código ao utilizador e verificar se o job existe.

Uma imagem com texto

Descrição gerada automaticamente

Figura 18 - DescobreJob

De seguida são utilizadas as funções **InsereNoOperacao** e **LeOperacao** (já mencionadas em cima).

### Remover operação de um determinado Job

Primeiramente é usada a função **DescobreJob** (mencionada acima) para descobrir se existe o job, e de seguida para remover uma operação desse job é utilizada a seguinte função:

* **RemoveOperacao:** Esta função irá pedir ao utilizador o código da operação que o utilizador deseja remover e depois de indicar o código irá mostrar a operação que foi removida.

Uma imagem com texto

Descrição gerada automaticamenteUma imagem com texto

Descrição gerada automaticamente

Figura 19 - RemoveOperacao

### Editar Operação de um determinado Job

No começo é pedido o código do job em que o utilizador pretenda editar uma operação usando a função **DescobreJob**, de seguidausada a seguinte função:

* **EditaOperação:** A função pede o código da operação que pretenda editar e de seguida o utilizador atualiza os dados ao seu gosto.

Uma imagem com texto

Descrição gerada automaticamente

Figura 20 - EditaOperacao

# Testes realizados

Durante o desenvolvimento do código foram feitos testes num ficheiro de texto. Nos testes da fase 1 foi adicionando primeiro as operações que seriam necessárias e de seguida as máquinas com o seu tempo.

Também foram feitos testes para as funções que determinam o máximo e o mínimo de tempo necessário para completar um Job, e por fim toda a informação é guardada no ficheiro de texto chamado dados.txt.

Para a fase 2 foram adicionados, removido e editado Jobs para além das operações que o programa já fazia.

Fazendo agora um teste, vamos adicionar um novo Job.

Primeiramente o programa abre o Menu, e será selecionado a opção 8:

Uma imagem com texto

Descrição gerada automaticamente

Figura 21 - Menu

De seguida pergunta se pretende adicionar as operações ao novo Job, vamos responder que sim.

Uma imagem com texto

Descrição gerada automaticamente

Figura 22 - Adicionar operações ao job

Depois de reponder que sim o programa pergunta quantas máquinas serão utilizadas na operação, respondemos 2.



Figura 23 - Maquinas utilizadas na operação

Respondemos os dados que queríamos e quando terminamos o programa pergunta se pretendo adicionar mais uma operação, respondemos Não.

Uma imagem com texto

Descrição gerada automaticamente

Figura 24 - Dados da operação

Adicionada a operação, fomos listar o Job criado e verificamos que este se encontra visível como o Job nº 9.

Uma imagem com texto

Descrição gerada automaticamente

Figura 25 - Job adicionado

Agora voltamos ao menu e selecionamos a opção “Inserir uma operação num Job” e agora vamos inserir os dados necessários.

Uma imagem com texto, eletrónica, teclado

Descrição gerada automaticamente

Figura 26 - Dados para atualizar

Como podemos verificar, a nova operação foi adicionada com sucesso!

Uma imagem com texto

Descrição gerada automaticamente

Figura 27 - Atualização de dados

De seguida vamos editar a operação nº 1, selecionamos no menu a opção 12 e o programa pergunta qual o id do Job em que pretendo alterar uma operação.

Selecionamos o Job nº 9 e a operação nº 1 tal como combinamos, e colocamos com os seguintes dados.

Uma imagem com texto

Descrição gerada automaticamente

Figura 28 - Operação editada

Listamos os Jobs e como podemos verificar a operação foi editada com sucesso!

Uma imagem com texto

Descrição gerada automaticamente

Figura 29 - Dados atualizados

De seguida vamos remover a operação Nº 2, no menu selecionamos a opção 11 e indicamos que queremos remover do job Nº 9 a operação Nº 2.

Uma imagem com texto

Descrição gerada automaticamente

Figura 30 - Remoção de uma operação

Ao fazer a listagem verifica-se que a operação foi eliminada com sucesso.

Uma imagem com texto

Descrição gerada automaticamente

Figura 31 - Listagem atualizada

Por fim, e para finalizar a nossa fase de testes, iremos remover o Job no qual fizemos o teste a todas as funcionalidades, o Job nº 9.

Depois de selecionar a opção 9 do menu, indicamos que queríamos remover o Job Nº 9 e esta foi a listagem após a remoção.

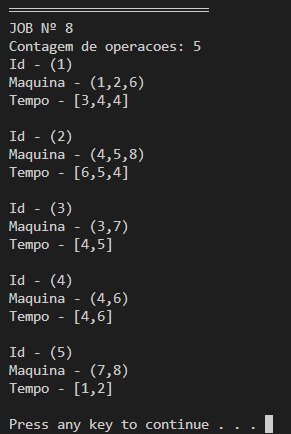


Figura 32 - Listagem final

# Conclusão

Podemos assim dar o projeto concluído com sucesso, onde atingimos todos os objetivos propostos. Consideramos um projeto interessante e desafiante, uma vez que conseguimos aperfeiçoar as capacidades desenvolvidas nas aulas, mas também aprender novas técnicas de programação em C, desenvolvendo o nosso conhecimento da linguagem para além do que nos é lecionado.

Por fim, agradeço a dedicação e a entrega que senti por parte do professor João Carlos Silva que sempre se demonstrou disposto a esclarecer-me e a enriquecer o meu conhecimento para assim ser possível a realização deste projeto.

# Bibliografia

Como bibliografia usei bastante os exercícios lecionados nas aulas e a internet, principalmente o StackOverFlow.