

UNIVERSIDADE TECNOLÓGICA DO PARANÁ
DAINF - DEPARTAMENTO ACADÊMICO DE INFORMÁTICA
BACHARELADO EM CIÊNCIAS DA COMPUTAÇÃO

GUSTAVO HENRIQUE CARDOSO DE ARAÚJO
MARLON HENRIQUE SANCHES

PROJETO ESTRUTURA DE DADOS I
SIMULAÇÃO DE PAGAMENTO DE MAQUINÁRIO

TRABALHO ACADÊMICO

PONTA GROSSA
2022

SUMÁRIO

TRABALHO ACADÊMICO	0
SUMÁRIO	1
Entrada de dados:	2
Saída de dados:	2
Funções Principais:	3
Cores:	3
Constantes:	3
Structs:	3
Funções Menu:	3
Funções Internas:	3
Funções de Leitura e Print:	4
Funções de Criação de Arquivos:	4
Funções de Simulação:	4
DEBUG FUNCTIONS:	5
Estruturas:	5
struct product	5
struct machine	6
struct headProduct	6
struct headMachine	7
struct line	7
struct machineOnProduction	8
struct packaging	8
Testes Executados:	9
TESTE 1:	9
TESTE2:	11
Comentários Gerais:	13

Entrada de dados:

- Máquinas: programa importa os dados das máquinas de um arquivo tabela nomeado "Maquinas.csv".
- Produtos: programa importa os dados dos produtos de um arquivos tabela nomeado "Produtos.csv".
- Simulação: o usuário é permitido através de um menu criar sua própria simulação podendo selecionar dentre as máquinas importadas e estipulando suas quantidades.

Saída de dados:

- Durante a simulação é exibido o progresso do pagamento e do ano passado.
- Após o término da simulação são exportadas (Resultados.csv) as informações completas dos custos, detalhados por produtos, consumo das máquinas e lucro.

Funções Principais:

O código é separado em diferentes regiões cada uma delas contendo funções.

Regiões:

Cores:

Aqui temos apenas as constantes relacionadas a cores que serão usadas para modificar cores do texto no terminal.

Constantes:

Em Constantes tem-se os valores *HardCoded* do código, ou seja, valores que serão usados durante o código e já são inseridos diretamente nele, no geral temos tempo de simulação, custo por **Kilowatt hora**, nomes dos arquivos de carregamento e de saída.

Structs:

Em *Structs* têm-se as estruturas principais do código que serão ilustradas no próximo tópico deste documento.

Funções Menu:

Em Funções Menu tem-se a função *creditsMenu*, a qual imprime o menu de créditos.

Funções Internas:

Em Funções Internas têm-se as funções utilizadas por outras funções, ou seja, funções que repassam valores, realizam cálculos ou alocam memória para outras funções.

Destas vale destacar as funções de alocação de nós das estruturas e a função *randomProduct* a qual gera um produto aleatório na proporção adequada conforme a tabela do PDF de instruções.

Funções de Leitura e Print:

Em Funções de Leitura e Print têm-se as funções que carregam dados de arquivos e escrevem informações na tela.

Destacam-se a *loadProducts* e a *loadMachines*, que carregam os dados das planilhas de Produtos e Maquinas. E a *progresPrint* que atualiza os valores na tela durante a simulação.

Funções de Criação de Arquivos:

Em Funções de Criação de Arquivos tem-se duas funções importantes:
createSimulationFile - Função a qual cria o arquivo de simulação que será carregado posteriormente para uma simulação.

createResultsFile - A qual retorna os valores pedidos no PDF de instrução sobre uma simulação que acabara de acontecer.

Funções de Simulação:

Em Funções de Simulação tem-se as principais funções do programa todo.

simulation - Esta função é responsável por começar a simulação, ela carrega as maquinas de simulação gera seu custo total e chama a *simulationLoop*.

simulationLoop - Basicamente é o coração deste programa (e o cérebro também), esta função roda segundo a segundo até completar os 2 anos ou até os ganhos superarem os custos (ou seja, começar a ter lucro). Ela é responsável ainda por chamar todas as outras funções desta região que possuem nomes auto-explicativos.

Outras funções desta região:

```
getCostByProductID
```

```
getGainByProductID
getDeteriorationTimeByProductID
getTimeOfProductionProductOnMachine
getMachineThatAcceptTypeOfProductANDHaveShortestList
addProductToMachineByID
machineTerminateProduction
removeExpiredProducts
updateTimeOfProductionOfMachines
updateDeteriorationTimeOfProducts
calculateMachinesCostHour
```

DEBUG FUNCTIONS:

Em *DEBUG FUNCTIONS* tem-se as funções usadas para debug durante o desenvolvimento do programa.

printMachinesOnProductionAndList - Imprime as maquinas em produção e a lista de produtos na fila de cada maquina.

Estruturas:

struct product

int ID
char productionType [2]
float productionCost
int deteriorationTime
int productionProbability
struct product *next

int ID -> identificador do Produto no programa

char productionType -> tipo de Produto podendo ser entre:

C - Coxinha

P - Peixe

A - Almôndega

float productionCost -> custo de produção do determinado produto
int deteriorationTime -> tempo de deterioração do determinado produto
int productionProbability -> probabilidade de entrada desse produto dado a entrada de um novo produto aleatório
struct product *next -> ponteiro indicador do próximo produto

struct machine

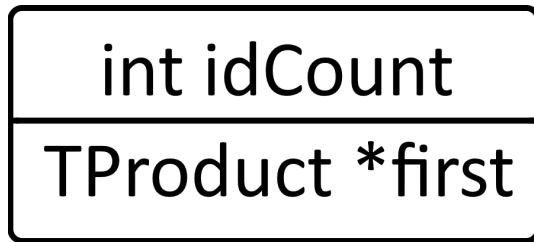
int ID
char model[30] char productionType[2] int productionTime[3] int consumption int price
struct machine *next

int ID -> identificador da Máquina no programa
char productionType -> tipo de Produto que a máquina pode empacotar podendo ser entre:

- C - Coxinha
- P - Peixe
- A - Almôndega
- T - Todos os produtos

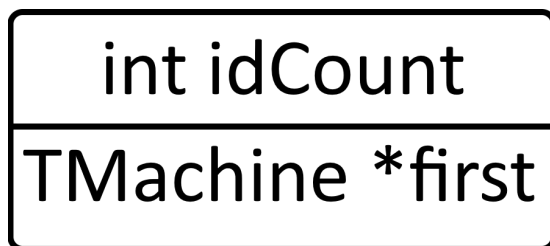
int productionTime -> tempo de produção para cada tipo de produto
int consumption -> consumo de KW/h da máquina
int price -> preço da máquina
struct machine *next -> ponteiro indicador da próxima struct machine

struct headProduct



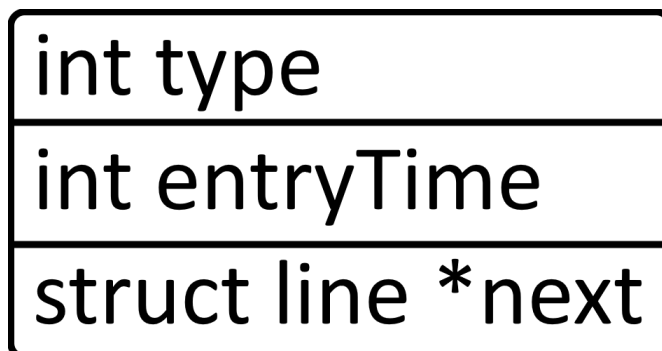
int idCount -> armazena a quantidade de produtos importados da tabela de produtos
TProduct *first -> aponta para o primeiro produto da lista

struct headMachine



int idCount -> armazena a quantidade de máquinas importadas da tabela de máquinas
TMachine *first -> aponta para a primeira máquina da lista

struct line



int type -> tipo de produto baseado no id
int entryTime -> tempo de entrada do produto na fila
struct line *next -> ponteiro apontando para a próxima struct line

struct machineOnProduction

int ID
char model[30] char productionType[2] char productionTime[3] int consumption int price TLine *first int numberOfProducts int timeOfProduction
struct machineOnProduction *next

int ID -> identificador da máquina

char model[30] -> nome da máquina

char productionType -> tipo de Produto que a máquina pode empacotar podendo ser entre:

C - Coxinha

P - Peixe

A - Almôndega

T - Todos os produtos

int productionTime -> tempo de produção para cada tipo de produto

int consumption -> consumo de KW/h da máquina

int price -> preço da máquina

TLine *first -> ponteiro para o primeiro produto da fila de produtos da máquina

int numberOfProducts -> número de produtos na fila de produtos da máquina

int timeOfProduction -> tempo de produção do produto em destaque

struct machineOnProduction *next -> ponteiro indicador da próxima struct machineOnProduction

struct packaging

int cBatches	int cWasted
int pBatches	int pWasted
int aBatches	int aWasted
int cProduction	int FinalTime
int pProduction	int totalCost
int aProduction	int totalGain

int cBatches -> quantidade total de lotes de Coxinha

int pBatches -> quantidade total de lotes de Peixe

int aBatches -> quantidade total de lotes de Almôndega

int cWasted -> quantidade de lotes desperdiçados de Coxinha

int pWasted -> quantidade de lotes desperdiçados de Peixe

int aWasted -> quantidade de lotes desperdiçados de Almôndega

int cProduction -> quantidade de lotes vendidos de Coxinha

int pProduction -> quantidade de lotes vendidos de Peixe

int aProduction -> quantidade de lotes vendidos de Almôndega

int finalTime -> tempo de fim da simulação

int totalCost -> custo total da simulação

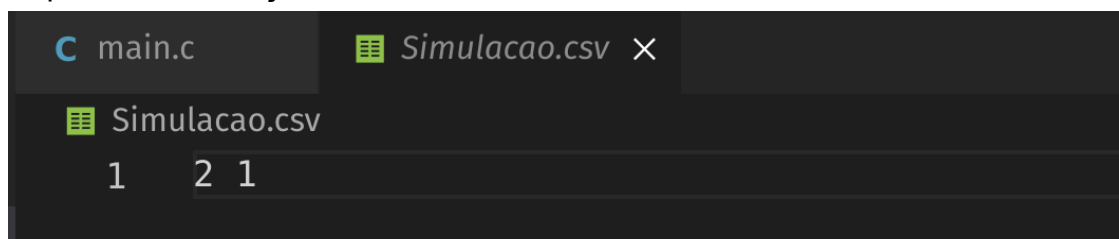
int totalGain -> lucro total da simulação

Testes Executados:

TESTE 1:

Uma maquina de ID 2, ou seja, uma ChickenPak.

Arquivo de Simulação:



```

main.c  Simulacao.csv X
Simulacao.csv
1  2  1

```

Programa após a execução da simulação:

```
Nº Loops: 63072000

< 100% >
-----
  \
   \
    .-.-.
    |o_o |
    |:_/ |
   //   \
  (|     |)
 / \   / \
 \   \   /
  \___/   \
      \___/

Tempo:
63072000 [=====] 100% 63072000
Dinheiro:
184418.25 [                                     ] 0% 22209045.61

Simulação finalizada com sucesso!
0 arquivo de resultados (Resultados.csv) foi criado com sucesso!
```

Arquivo de Resultado:

```
C main.c  Resultados.csv x
Resultados.csv
1  Tempo_simulação: 63072000
2
3  X Coxinha Peixe Almondega Total
4  Lotes_Empacotados 127185 0 0 127185
5  Ganhos 184418.25 0.00 0.00 184418.25
6  Lotes_Perdidos 15639159 0 0 15639159
7  Custo 12613088.00 6621303.50 2524254.00 21758646.00
8
9
10 MAQUINAS
11 ID Modelo Custo
12 1 ChickenPak 350400
13
```

Visualização em planilha do arquivo de Resultado:

	A	B	C	D	E
1	Tempo simulação:	63072000			
2					
3	X	Coxinha	Peixe	Almondegas	Total
4	Lotes_Empacotados	127185	0	0	127185
5	Ganhos	184418.25	0	0	184418.25
6	Lotes_Perdidos	15639159	0	0	15639159
7	Custo	12613088	6621303.5	2524254	21758646
8					
9					
10	MAQUINAS				
11	ID	Modelo	Custo		
12		1 ChickenPak	350400		
13					
14					

Percebe-se que apenas com uma maquina a simulação é parada pelo o tempo (2 anos) 63072000 segundos, pois uma maquina ainda mais sendo apenas de coxinha como o exemplo não dará conta da produção.

TESTE2:

Arquivo de Simulação:

```

C main.c  Simulacao.csv x
Simulacao.csv
1 5 2
2 4 3
3 3 5
4 1 1

```

Programa após a execução da simulação:

```
Nº Loops: 5532615

< 8% >
-----
  \
   \
    .-.
   |o_o|
   |: / |
  //  \ \
 (|    |)
 /' \  \ \
 \    /  \
  \____/

Tempo:
5532615 [====] 8% 63072000
Dinheiro:
3522775.88 [=====] 100% 3522775.71

Simulação finalizada com sucesso!
0 arquivo de resultados (Resultados.csv) foi criado com sucesso!
```

Arquivo de Resultado:

```
Resultados.csv
1  Tempo_simulação: 5532615
2
3  X Coxinha Peixe Almondega Total
4  Lotes_Empacotados 1278913 612568 554020 2445501
5  Ganhos 1854423.88 1225136.00 443216.00 3522776.00
6  Lotes_Perdidos 105192 215604 0 320796
7  Custo 1107291.25 579721.06 221608.41 1908620.62
8
9
10 MAQUINAS
11 ID Modelo Custo
12 1 EnSacAll 61440
13 2 EnSacAll 61440
14 3 Plastific 53760
15 4 Plastific 53760
16 5 Plastific 53760
17 6 AllPak 33792
18 7 AllPak 33792
19 8 AllPak 33792
20 9 AllPak 33792
21 10 AllPak 33792
22 11 FishPak 30720
23
```

Visualização em planilha do arquivo de Resultado:

	A	B	C	D	E	F
1	Tempo_simulação:	5532615				
2						
3	X	Coxinha	Peixe	Almondega	Total	
4	Lotes_Empacotados	1278913	612568	554020	2445501	
5	Ganhos	1854423.88	1225136	443216	3522776	
6	Lotes_Perdidos	105192	215604	0	320796	
7	Custo	1107291.25	579721.06	221608.41	1908620.62	
8						
9						
10	MAQUINAS					
11	ID	Modelo	Custo			
12		1 EnSacAll	61440			
13		2 EnSacAll	61440			
14		3 Plastific	53760			
15		4 Plastific	53760			
16		5 Plastific	53760			
17		6 AllPak	33792			
18		7 AllPak	33792			
19		8 AllPak	33792			
20		9 AllPak	33792			
21		10 AllPak	33792			
22		11 FishPak	30720			
23						

Percebe-se que aqui, com uma configuração mais balanceada de máquinas, a simulação acabou em 8% do tempo máximo de 2 anos (5532615 segundos totais).

Comentários Gerais:

O código atende tudo que é pedido, também foi feito de tal forma que pode ser facilmente modificado e adaptado. Assim, caso queira-se adicionar mais máquinas, modificar seus valores ou modificar valores de produtos basta alterar as planilhas e em alguns casos poucas partes do programa (como a função *randomProduct* para caso alteração na porcentagem do aparecimento dos produtos na fila de empacotamento).

Ainda se destaca a fácil automatização deste programa por poder receber entradas e realizar saídas em arquivos facilitando a outros programas o utilizar como motor (back-end) de simulação.

Por último as funções que mais deram trabalho neste código foram as de simulação, pois além de ser o objetivo do código o programador tem que se atentar a todos os detalhes para a simulação.