Trabalho de AEDA

Empresa de Transportes e Mercadorias

Componentes:

Allan Borges de Sousa	201800149
Amanda de Olivera Silva	201800698
Juliane de Lima Marubayashi	201800175

DESCRIÇÃO DO PROBLEMA

O problema é dividido em basicamente em três vertentes:

1) Adicionar ao programa uma forma de gerenciar de funcionários (motoristas). Assim, ainda deve ser computado o número de horas trabalhadas e a alocação de funcionários para serviços deve ser automática.
2) Verificação de inatividade de clientes, ou seja, clientes com mais de um ano sem requisitar serviços são considerados inativos. Deve ser possível, principalmente, pesquisar e visualizar estes.
3) Possibilitar o gerenciamento de oficinas de reparo de camiões. Tais oficinas efetuam serviços e possuem determinada especialidade de marca de automóveis. Ainda, a cada vez que uma oficina aceita um pedido, sua disponibilidade diminui.

DESCRIÇÃO DA SOLUÇÃO

1) Os motoristas são gerenciados por uma **BST ordenada por horas de trabalho** composta pela classe ***Motorista*.** A ordenação foi possível pela implementação do operador <, o qual considera a ordem crescente de horas e ordem crescente de nif respectivamente. Para o gerenciamento da BST é feita pela classe Workers.
2) A inatividade de clientes é gerenciada por uma **Hash Table com *Clientes.** Como os clientes inativos depende da data do último serviço, foi adicionado à classe ***Clientes*** um parâmetro do tipo ***Data*** que armazena a data do último pedido. Assim, sempre que *data atual - data último pedido >= 1 ano* o cliente é considerado inativo. Novos clientes possuem data default = 1500/01/01, no formato yyyy/mm/dd.

   Foram utilizados overload do operador () para indicação da igualdade de clientes (nif cliente 1 == nif cliente 2) e para a criação da função hash, que nos caso é o valor do nif.

DESCRIÇÃO DA SOLUÇÃO

3) Uma oficina é representada pela classe ***Workshop,*** composta por nome, uma lista de marcas nas quais o workshop se especializa e o tempo até que esse workshop possa realizar um novo serviço. Os workshops da empresa são armazenados num vetor na classe ***Empresa*** devido a necessidade de eventual acesso aleatório e em uma priority queue. A **priority queue** da stl , como é max-heap, está ordenada em ordem decrescente de indisponibilidade. Quando camião pede um serviço, é atualizado o workshop no topo da fila. Quando o serviço é específico é criada uma priority queue auxiliar, procura-se um workshop que possua a marca do veículo como especialidade e a priority queue original é atualizada. Para a organização da **priority queue** é usada o overload operator <.

Toda vez que um serviço é requisitado a indisponibilidade de todos os workshops diminui em um, mas a indisponibilidade do workshop requisitado é aumentada. Quanto menor a indisponibilidade, mais “livre” o workshop está.

ESTRUTURA DE FICHEIROS

Motoristas.txt

```
1  Worker1
2  2
3  0
4  Worker3
5  3
6  0
```

Trabalhador
NIF
Horas trabalhadas

camioes.txt

```
1  MARCA1
2  23
3  Congelado
```

Marca
Carga Máxima
Tipo

clientes.txt

```
1  Cliente2
2  201800149
3  1500/01/01
4  Cliente1
5  201800175
6  1500/01/01
```

Nome
NIF
Data do último serviço
OBS:
Data no formato yyyy/mm/dd

workshops.txt

```
1  Teste 2
2  2
3  BMW
4  TOYOTA
5  20
6  Teste 3
7  2
8  HONDA
9  YAMAHA
10 30
```

Nome
Quant. de marcas
Marca1
Marca2
Indisponibilidade

servicosx.txt

```
1  Rio de Janeiro
2  41
3  42
4  Petropolis
5  40
6  40
7  Perigoso
8  201800177
9  40
10 Quebravel
11 3
```

Saída
Latitude
Longitude
Destino
Latitude
Destino
Tipo
Cliente
Carga máxima
Característica
Camiões

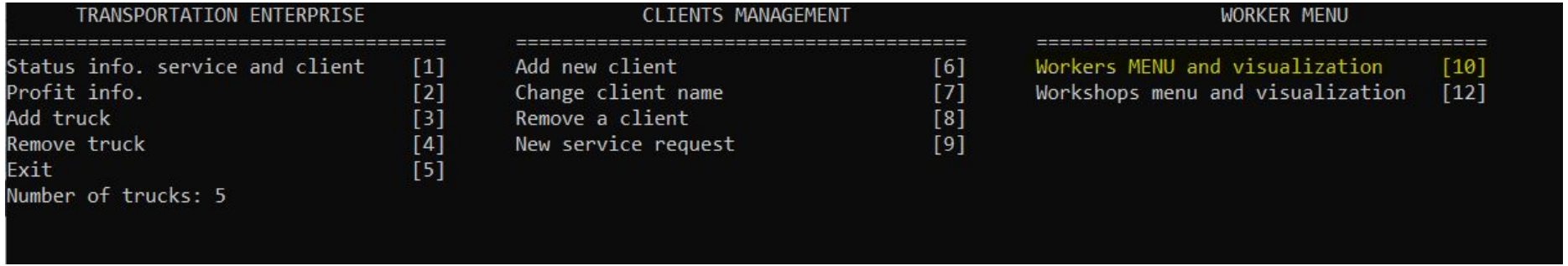
taxas.txt

```
1
2
3  Perigoso
4  43
5  Congelado
6  30
```

Linha em branco
Linha em branco
Nome da taxa
Valor
Nome da taxa
Valor

LISTA DE FUNCIONALIDADES - Motorista

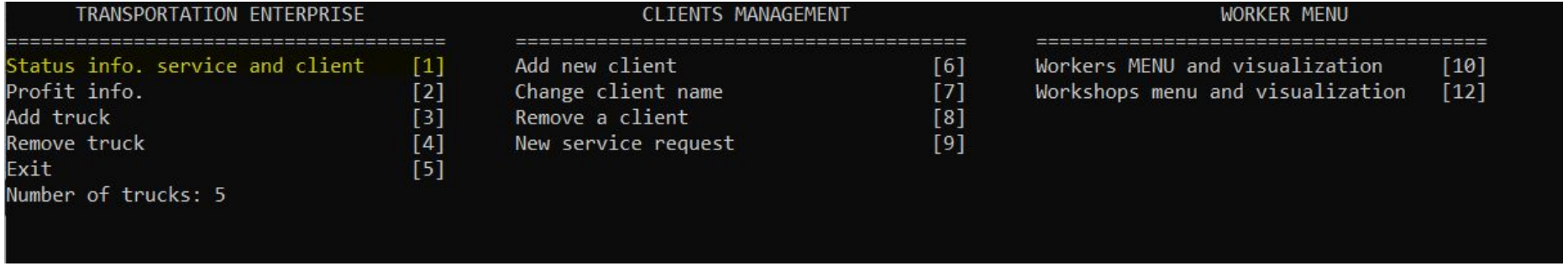
As funcionalidades relacionadas a motoristas podem ser acedidas a partir do menu principal opção 10



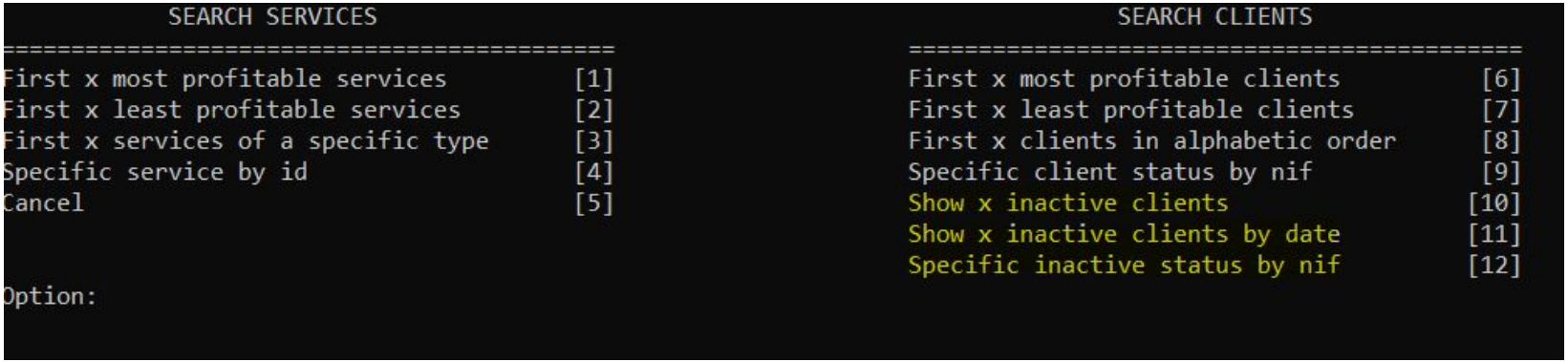
Todas as funcionalidades citadas foram completamente implementadas:

LISTA DE FUNCIONALIDADES - clientes inativos

Todas as funcionalidades dos clientes inativos podem ser acedidas a partir do menu principal, opção 1

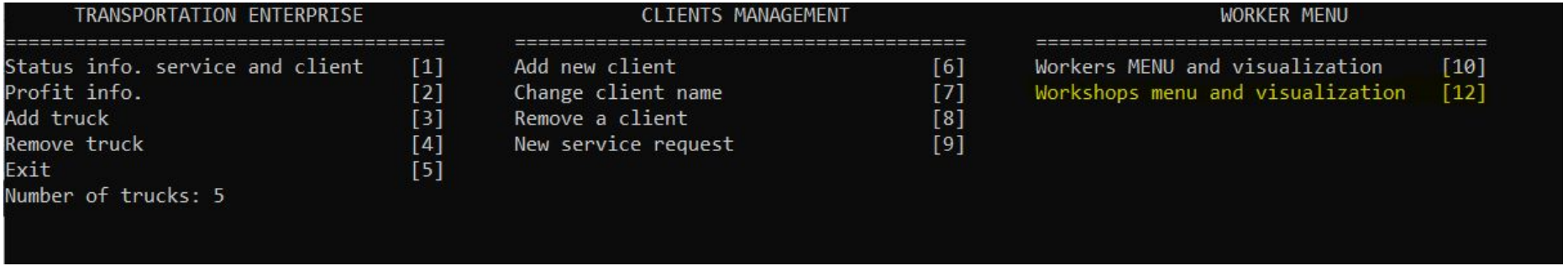


Todas as funcionalidades citadas foram completamente implementadas:

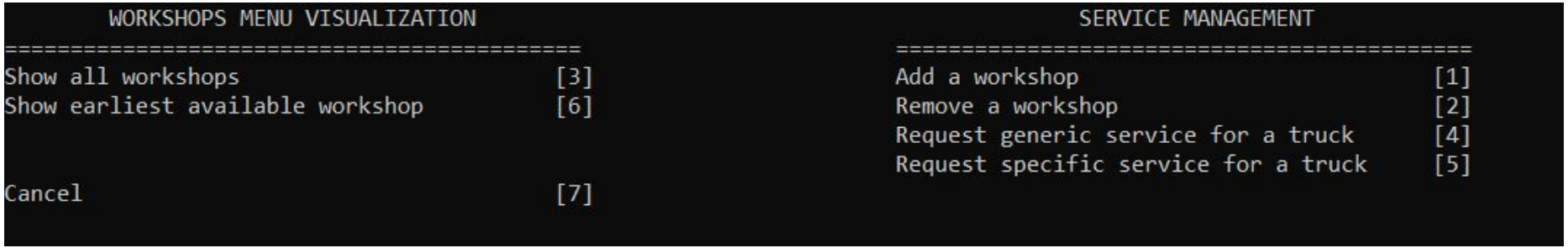


LISTA DE FUNCIONALIDADES - Workshop

As funcionalidade dos workshops podem ser acedidas a partir do menu principal, opção 12:



Todas as funcionalidades citadas foram completamente implementadas:



DESTAQUE DE FUNCIONALIDADE

Gostaríamos de destacar a funcionalidade de printar motorista por ordem decrescente de horas:

Para a organização da bst em ordem decrescente de horas, a bst.h (arquivo source da estrutura) foi alterado. Houve a implementação de uma nova função chamada ***void getReversedTree(vector<Comparable>& v) const;*** que percorre recursivamente a árvore na ordem reversed pre-order.

```
324 template <class Comparable>
325 void BST<Comparable>::getReversedTree( vector<Comparable> & v) const{
326     getReversedTree_aux(this->root, v);
327 }
328
329 template <class Comparable>
330 void BST<Comparable>::getReversedTree_aux(BinaryNode<Comparable> * t, vector<Comparable> & v) const{
331     if (t != NULL){
332         getReversedTree_aux(t->right, v);
333         v.push_back(t->element);
334         getReversedTree_aux(t->left, v);
335     }
336 }
```

```
88  void Workers::printBST_reversed(int n){
89      int counter = 0;
90      vector<Motorista> m;
91      BST_Workers.getReversedTree(m);
92      for (auto const& it: m){
93          counter ++;
94          cout << it;
95          if (counter == n) break;
96      }
97  }
```

PRINCIPAIS DIFICULDADES E ESFORÇO

Neste trabalho, cada componente trabalhou de maneira igual.

As grandes dificuldades do trabalho estão relacionadas em:

- Pensar como os problemas seriam solucionados;
- Na divisão de tarefas dos componentes;
- Aprendizado do sistema de branchs do github;
- E principalmente em lidar com erros de escrita em arquivos herdados do projeto anterior e devido a diferença de sistema do grupo. No windows por exemplo, o “enter” é demarcado com “\r\n”, já no linux é usado apenas “ \n” e isto desencadeou uma série de erros na formatação dos arquivos e na leitura.