



UNIVERSIDADE FEDERAL DE SANTA CATARINA- UFSC
ENGENHARIA ELETRÔNICA

LUCAS ARMANDO CIELLO

Relatório projeto final

Florianópolis, 2018

LUCAS ARMANDO CIELLO

Relatório técnico apresentado como requisito parcial para obtenção de aprovação na disciplina Programação C++ para sistemas embarcados, no Curso de Engenharia Eletrônica, na Universidade Federal de Santa Catarina.

Prof. Dr. Eduardo Augusto Bezerra

Florianópolis, 2018

Sumário

Sumário	2
Resumo	4
Diagrama de Classes	5
Maquina de estado	5
Filas	5
Lista de disponibilidades.....	6
Maquina de estado.....	7
Funcionamento	7
Explicação Código	8
Sequencia de testes	8
Fila de anúncios	8
Explicação Código	9
Modelagem do modulo	10
Sequencia de testes	10
Funcionalidade adicional do sistema.....	10
Explicação Código	11
Modelagem do modulo	12
Sequencia de testes	12
Projeto final	12
Classe interface	13
Explicação do código	13
Arquitetura do sistema	13
Ferramentas utilizadas	14
Plano de testes.....	15
Dificuldades encontradas	16

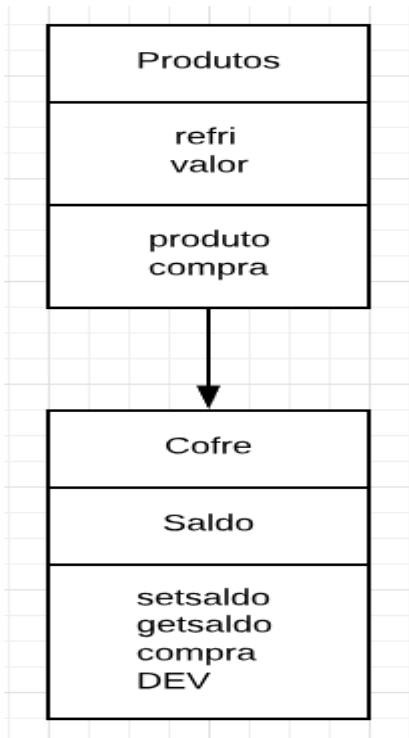
Apêndice	17
Manual do usuário	17

Resumo

Este relatório apresenta o diagrama de classes, o funcionamento da maquina de estado, as filas que serão utilizadas no projeto, a descrição do funcionamento adicional da maquina de refrigerante e o projeto final onde juntaremos todos esses funcionamentos. Será apresentado também a modelagem do sistema, arquitetura do sistema, ferramentas utilizadas, planos de testes e a apresentação e explicação dos seus códigos em C++.

Diagrama de Classes

Maquina de estado



O Diagrama é referente a maquina de estados, onde temos as classes produtos onde são armazenados os dados dos refrigerantes, bem como a classe cofre que é responsável pela gestão do dinheiro e da liberação do produto.

A classe cofre recebe a classe produto, tendo acesso aos seus atributos e métodos.

Filas

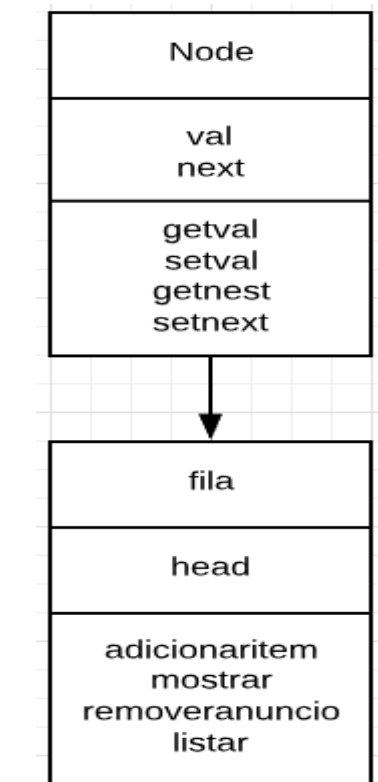


Diagrama referente às filas que serão utilizadas para implementação da dinâmica de propagandas.

A classe fila recebe a classe Node, tendo acesso aos seus atributos e métodos.

Lista de disponibilidades

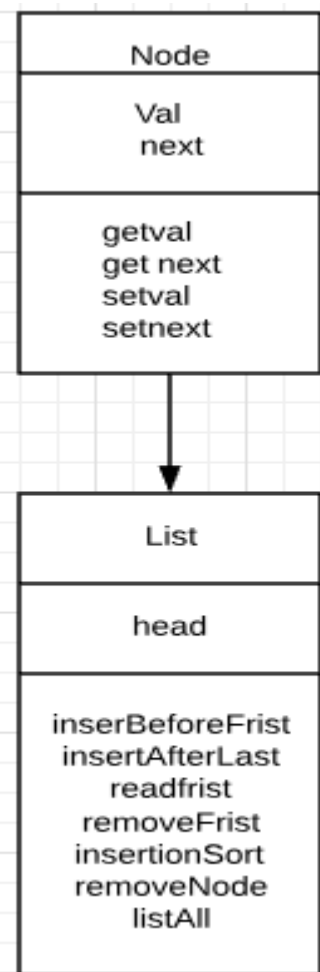


Diagrama referente lista de disponibilidades que serão utilizadas para implementação da funcionalidade extra do projeto.

A classe List recebe a classe Node, tendo acesso aos seus atributos e métodos. Tanta a classe List quanto a classe fila recebem a mesma classe Node, esta situação podemos perceber o conceito de herança, a classe Node é pai e as classes List e fila são filhas.

O programa final recebe as classes da Máquina de estado, da fila bem como as classes utilizadas para a funcionalidade extra do projeto, possibilitando a implementação de um programa que fará toda a gestão da máquina de refrigerante e em paralelo serão apresentadas as propagandas cadastradas pelo utilizador e a funcionalidade extra. Existem duas interfaces, uma para o cliente outra para o operador. Na interface do cliente existem botões para escolha dos refrigerantes e para simular as moedas e leds para representar a saída de refrigerante e das moedas, na interface do operador vão existir um teclado e um monitor, nesta interface será possível entrar no modo programador, que possibilita configurar os anúncios, os produtos cadastrados e sua disponibilidade.

Maquina de estado

A máquina fornece dois tipos de refrigerantes, MEET e ETIRPS, que estão disponível para ser escolhidos no menu do programa. Ambos refrigerantes custam R\$1,50 e a máquina reconhece moedas de R\$1,00, R\$0,50 e R\$0,25, no código o usuário indica a moeda que está colocando, visto que nesta etapa não será utilizado nenhum tipo de sensor ou botão. A maquina devolve automaticamente o troco quando o saldo da maquina passa de R\$1,50, além disso, durante a compra, o usuário pode desistir da transação, a opção DEV devolve as moedas inseridas até o momento. Somente após acumular um crédito de R\$1,50 o usuário pode obter um refrigerante.

Funcionamento

Estado atual	Comando de Entrada						
	Nada	M025	M050	M100	DEV	MEET	ETIRPS
S000	S000	S025	S050	S100	S000	S000	S000
S025	S025	S050	S075	S125	S000, D025	S025	S025
S050	S050	S075	S100	S150	S000, D050	S050	S050
S075	S075	S100	S125	S150, D025	S000, D075	S075	S075
S100	S100	S125	S150	S150, D050	S000, D100	S100	S100
S125	S125	S150	S150, D025	S150, D075	S000, D125	S125	S125
S150	S150	S150, D025	S150, D050	S150, D100	S000, D150	S000, DMEET	S000, DETIRPS

Através dessa tabela podemos verificar todas as possibilidades e o comportamento da maquina de estado.

Explicação Código

A classe produto é utilizada para armazenar os dados dos refrigerantes, a classe cofre é filha da classe produto, tendo acesso a seus atributos e métodos. A classe cofre é responsável por cuidar do saldo e da liberação do refrigerante, Ambas as classes têm seus construtores que colocam o programa em estado inicial. A função setsaldo é responsável por adicionar o saldo sempre que o usuário informar que uma moeda foi inserida na máquina, adicionando o saldo anterior ao valor da moeda adicionada, sempre que o saldo ultrapassar o valor de R\$1,50, a função devolve o saldo excedente. A função compra faz um teste para verificar se o saldo é de R\$1,50, caso seja, ela libera o refrigerante escolhido e o valor do refrigerante é descontado do saldo. A função DEV devolve o saldo acumulado na máquina, caso ele exista. Este código é preliminar, podendo ser alterado conforme a necessidade ao longo do projeto.

Sequencia de testes

Para comprovar o funcionamento da máquina de estado, foram realizados os seguintes testes:

- Tentamos adquirir o refrigerante sem saldo e com todas as opções de saldo diferentes de R\$ 1,50, ao tentar comprar o refrigerante sem o saldo disponível a máquina não responde ao comando.
- Tentamos inserir variáveis diferentes das definidas para as moedas, sempre que é lida uma variável diferente da esperada é apresentada a mensagem “objeto não identificado”.
- Tentamos utilizar a função DEV para devolver dinheiro sem ter dinheiro na máquina, a máquina não responde a esse comando sem ter saldo.

Fila de anúncios

O sistema é composto por duas filas, a fila 2 é utilizada para armazenar os anúncios, enquanto a fila 1 é utilizada para mostrar os anúncios. Sempre que um anúncio novo é adicionado, ele é adicionado na fila 2, quando o usuário quiser apresentar este anúncio ele carregar os anúncios incluídos na fila 2 na fila 1, então a fila 1 apresenta os anúncios atualizados. Sempre que um anúncio é apresentado ele é removido da fila e adicionado no fim da fila, no caso onde os anúncios são carregados de uma fila para outra, o anúncio é adicionado no fim da fila 1 e removido da fila 2.

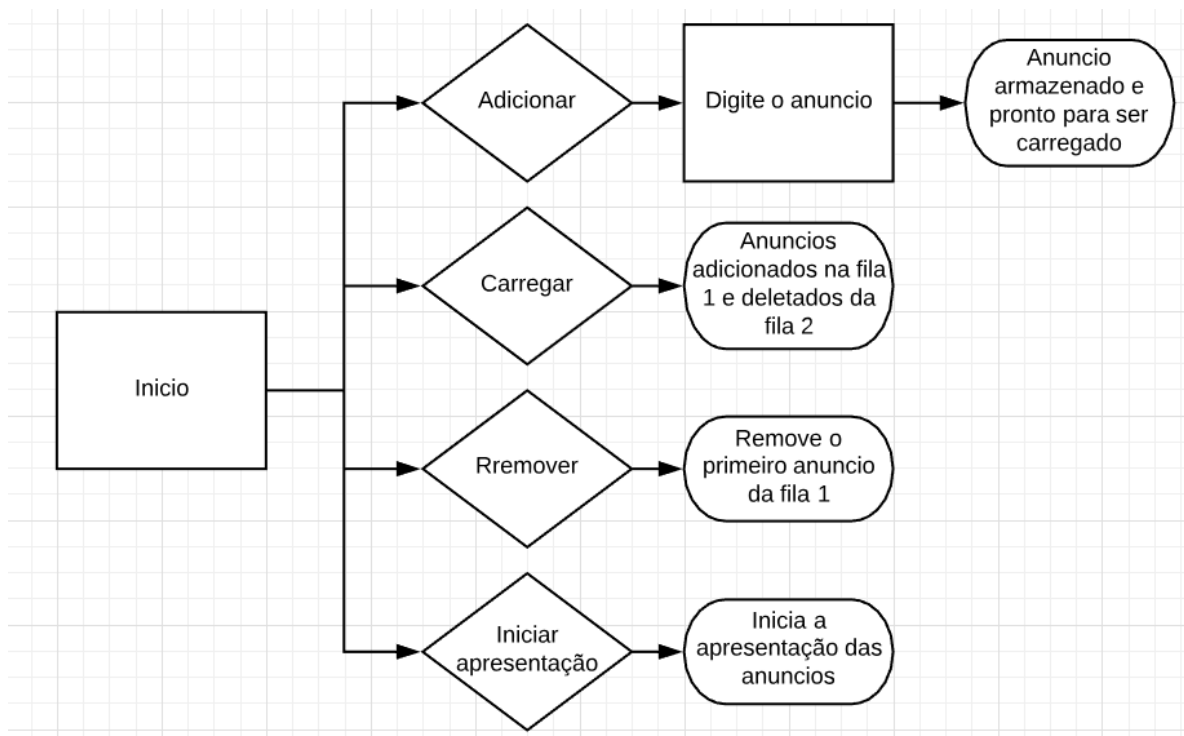
Explicação Código

A classe nodo é utilizada para armazenar os anúncios e um ponteiro que aponta para o próximo nodo, a classe tem seu construtor que é utilizado para armazenar os valores nas variáveis, bem como as funções getVal e getNext que são utilizados para apresentar o anúncio do nodo em questão e o próximo nodo respectivamente, as funções setVal e setNext são utilizadas para definir qual será o anúncio no nodo e qual será o próximo nodo.

A classe fila recebe a classe Node, tendo acesso aos seus atributos e métodos, possui seu construtor e destrutor, bem como as funções, adicionaritem que é responsável por adicionar um novo anúncio, a função mostrar que retorna o anúncio, a função removeranuncio que remove o anúncio da fila e a função listar, que lista todos os anúncios adicionados na fila, essa função não será utilizada no código, pois foge dos conceitos de fila, ela foi utilizada para testes no programa.

Na main é apresentado um menu para o operador poder adicionar, carregar, remover anúncios. Para cada função e funcionalidade do sistema será implementada uma outra função que possibilita o código ser utilizado na plataforma Raspberry PI, utilizando o conceito de polimorfismo para alterar a plataforma a ser utilizada. Os anúncios serão apresentados em um display no Raspberry PI ou na própria tela. O código apresentado é uma versão inicial e será alterado para atender os requisitos do sistema desejado.

Modelagem do modulo



Sequencia de testes

Para comprovar o funcionamento das filas de anuncio, foram realizados os seguintes testes:

- Tentamos carregar os anúncios sem haver anúncios salvos, o código não responde a esse comando, assim que se tentar carregar sem ter anúncios ele volta para o menu inicial.
- Tentamos remover anúncios sem haver anúncios cadastrados, o código não responde a esse comando, ao realizado ele volta para o menu inicial.

Funcionalidade adicional do sistema

A funcionalidade extra consiste em apresentar ao cliente as bebidas que podem ser vendidas na maquina e se estão disponíveis para a compra no momento

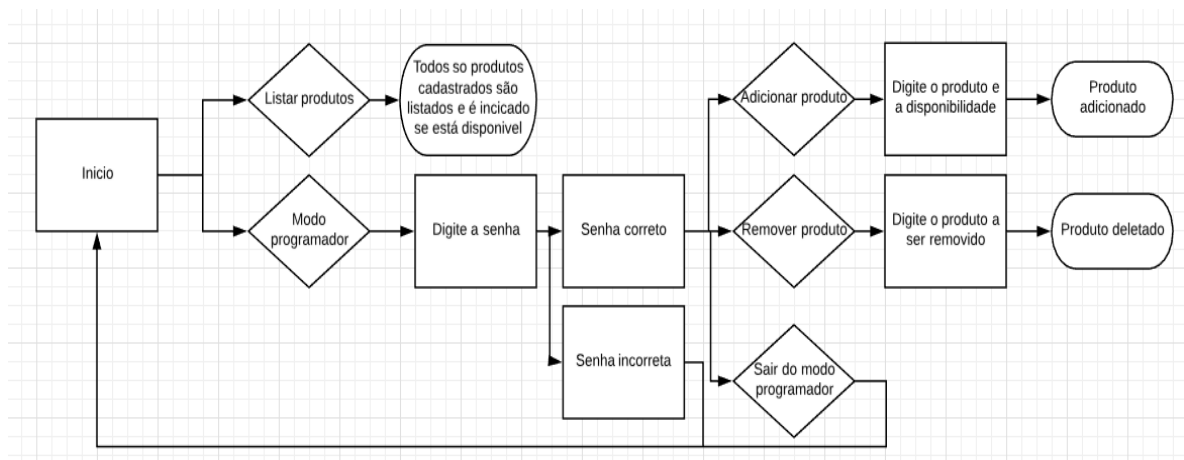
ou não. Para essa implementação será utilizado a estrutura lista. Existirá um botão com a opção de listar as bebidas, então todas as bebidas que estão cadastradas na maquina serão apresentadas para o cliente, também é possível entrar no modo programador, que dá a liberdade para o operador de cadastrar novos produtos, remover produtos e alterar produtos já cadastrados. Também será implementado o modo programador, que será utilizado no projeto final.

Explicação Código

A classe List recebe a classe Node, tendo acesso aos atributos e métodos já explicados da classe Node. A classe List possui as funções, `insertBeforeFrist` que é utilizada para adicionar novos produtos antes de todos os item já cadastrados, o item cadastrado por essa função será o primeiro item da lista, a função `insertAfterLast` é utilizada para adicionar novos produtos após os item já cadastrados, o item cadastrado por esta função será o ultimo da lista, se um produto for adicionado por uma das funções descritas quando a lista estiver vazia, ele será o primeiro da lista, a função `headFrist` é utilizada para retornar o primeiro produto adicionado na lista, a função `removeFrist` é utilizada para deletar o primeiro item da lista, a função `removeNode` é utilizada para deletar um item específico da lista, é passado o produto a ser deletado, a função procura esse produto e deleta o mesmo.

Na main é feito um menu onde o usuário pode listar, onde é utilizado a função `listAll` ou entrar no modo programador, para acessar o modo programador é solicitado uma senha, a senha é 9090, caso o usuário digite a senha errada o programa volta para o menu inicial, caso a senha esteja correta, é apresentado um novo menu para o usuário navegar, onde é possível acessar as opções, adicionar produto, onde é utilizado a função `insertBeforeLast` para adicionar um novo produto a lista, a opção remover produto, onde o usuário deve passar o anúncio em seguida, a função `removenode` procura o anúncio e o deleta, e a opção sair do modo programador, onde o programa volta ao menu inicial.

Modelagem do modulo



Sequencia de testes

Para comprovar o funcionamento da funcionalidade extra, foram realizados os seguintes testes:

- Tentamos listar os produtos sem haver produtos cadastrados, o código não responde a esse comando, ao realizado ele volta para o menu inicial.
- Tentamos digitar a senha errada para acessar o modo programador, é apresentada a mensagem “senha invalida” e volta para o menu inicial.
- Tentamos remover produtos sem haver produtos cadastrados, o código não responde a esse comando, ao realizado ele volta para o menu inicial.

Projeto final

O projeto final apresenta as três funcionalidades apresentadas anteriormente, para poder apresentar o sistema tanto no computador quanto no Raspberry pi, foi criado a classe interface, onde é possível alterar onde o projeto será apresentado

Classe interface

Na classe interface é possível verificar que existem diversas partes comentadas, para poder apresentar o código no computador é preciso comentar partes que são necessárias na apresentação no Raspberry, como por exemplo, a declaração de pinos, o uso de botões e leds. Na apresentação do computador para a interface é feita com cout e cin, tudo apresentado na própria tela do computador, já para a apresentação no Raspberry são utilizados leds e botões, os botões representam as entradas, como por exemplo, para escolher um refrigerante, para simular que esta sendo colocadas moedas na maquina, para apresentar os anúncios, devolver saldo, listar disponibilidades e ate entrar no modo programador.

Explicação do código

No código é incluído a biblioteca wiringPi.h que é responsável pela gestão dos pinos do Raspberry, são declarados os pinos e em seguida configuramos se cada pino é de entrada ou saída. A classe interface tem as funções, menuinicialpc que é responsável por apresentar o menu inicial no computador, armazenar a escolha do cliente e retornar essa escolha para o código principal, a classe menuinicialrasp tem o mesmo objetivo, entretanto nesta opção são usados botões conectados a placa para a leitura de dados, da mesma maneira acontece nas funções menumoedapd e menumoedarasp, comprapc e comprarasp e devpv e devrasp. Na main existe um ponteiro p, que apontada para a função do sistema correspondente, dependendo da interface a ser utilizada.

Todos os códigos e versões estão disponíveis em <https://github.com/lucasciello/Projeto-final>

Arquitetura do sistema

Para as três funcionalidades utilizam a estrutura de classes que nos possibilitou a proteção dos dados, por exemplo, a classes cofre é responsável por

fazer a gestão do dinheiro armazenado na maquina, somente a classes cofre pode alterar ou ver algo relacionado ao dinheiro na maquina, também nas três funcionalidades utilizamos o conceito de herança, que nos possibilitou a otimização do código, por exemplo, as classes fila e lista são filhas da classe node, através da herança foi possível desenvolver somente uma classe node que é utilizada nas classes fila e lista. Para a implementação da apresentação dos anúncios utilizados a estrutura de fila, na implementação da apresentação dos produtos disponíveis, foram utilizados as estruturas de fila e lista, respectivamente, isso nos possibilitou fazer a gestão de todas as variáveis de modo muito mais fácil e organizado. Também foram utilizadas estruturas como if, swile, for, que possibilitaram implementar a logica do funcionamento do sistema. Utilizaremos também o conceito de funções virtuais e polimorfismo, para poder utilizar o código tanto no computador quanto em um Raspberry PI, de modo que para cada uma das interfaces existirá uma função adequada, quando o código for utilizado no computador, a variável correspondente deve apontar para a função do funcionamento no computador, quando for utilizado no Raspiberry PI, a variável deve apontar para a função do Raspiberry PI, essa função existira na classe pai como uma função virtual, e também existira nas classes filhas, sendo para cada uma das classes filhas uma interface diferente. O uso da programação orientada a objeto nos permite o uso de todos esses recursos, facilitando e melhorando os códigos.

Ferramentas utilizadas

Para implementação do projeto serão utilizados os conceitos vistos em aula, herança, friends, template, funções virtuais, sobrecarga de operador, polimorfismo, para fazer com que as filas possam ser utilizadas em paralelo, uma apresentando os anúncios e outra adicionando novos anúncios, será utilizado o conceito de threads. Na parte física do projeto será utilizado um display para apresentar as mensagens de anúncios, botões para escolher os refrigerantes bem como para simular as moedas, cada moeda será representada por um botão, para o modo de programação será utilizado um teclado para poder digitar a senha e navegar entre as opção, também será utilizado um monitor para visualização. Dentre outros

métodos e componentes que serão necessários para o funcionamento do projeto e a sua implementação na plataforma Raspberry PI.

Plano de testes

O projeto final será apresentado no computador, para a apresentação iremos seguir o seguinte roteiro de testes:

1. Comprar os refrigerantes com saldo R\$ 0,00, ao iniciar o código iremos apertar 1 e Enter em seguida 2 e Enter.
2. Adicionar um saldo na maquina de R\$ 2,00, serão adicionados as seguintes moedas R\$ 0,25, R\$ 0,25, R\$ 0,50 e R\$ 1,00. Apertar 3 e Entrar para entrar no menu de moedas apertar 1 e Enter para selecionar a moeda de R\$ 0,25, repetir o para a moeda de R\$ 0,25, e realizar o procedimento novamente para as moedas de \$ 0,50 e R\$ 1,00.
3. Comprar os refrigerantes com saldo R\$ 1,50, após fazer a adição das moedas apertar 1 e Enter para comprar o refrigerante MEET.
4. Adicionar um saldo qualquer a maquina, apertar 4 e Enter para devolver o saldo adicionado.
5. Apertar 5 e Enter para a apresentação de anúncios, como a maquina ainda não foi configurada, ainda não existem anúncios.
6. Apertar 6 e Enter para apresentar os produtos cadastrados na maquina, como a maquina ainda não foi configurada, existem apenas os produtos iniciais.
7. Apertar 7 e Enter, 9090 e Enter para entrar no modo programador, aperta 1, Enter e digitar um anúncio (sem espaço). Repetir o processo para pelo menos 3 anúncios.
8. Apertar 7 e Enter, 9090 e Enter para entrar no modo programador, aperta 2 e Enter carregar os anúncios da fila de espera para a fila de reprodução.

9. Apertar 7 e Enter, 9090 e Enter para entrar no modo programador, aperta 3 e Enter para deletar o primeiro anuncio da fila.
10. Apertar 7 e Enter, 9090 e Enter para entrar no modo programador, aperta 4, Enter e digitar o nome do produto e sua disponibilidade como indicado (sem espaço). Repetir o processo para pelo menos 3 produtos.
11. Apertar 7 e Enter, 9090 e Enter para entrar no modo programador, aperta 5, Enter e digitar o anuncio e sua disponibilidade no padrão indicado, para deletar o anuncio.
12. Apertar 7 e Enter, 9090 e Enter para entrar no modo programador, aperta 6 e Enter para sair do modo programador.
13. Apertar 5 e Enter para a apresentação de anúncios, serão apresentados os anúncios cadastrados.
14. Apertar 6 e Enter para apresentar os produtos cadastrados na maquina, serão apresentados os produtos cadastrados.

Dificuldades encontradas

Para implementar um código que possibilite a apresentação das mensagens de anuncio na fila 1 e em paralelo poder adicionar ou remover anúncios da fila 2, tentamos utilizar o conceito de threads, conseguimos entender o seu conceito e fazer testes simples, apresentando mensagens definidas na própria threads, entretanto, ao tentar passar a informação da fila 1 para a threads, não obtivemos sucesso. No primeiro momento tentamos implementar a funcionalidade com as threads na própria main, após inúmeras tentativas tentamos criar uma classe que faria a gestão das threads, também sem sucesso. Como as mensagem não puderam ser apresentadas em paralelo com o resto do funcionamento do código, também tivemos restrições na questão do botão que faria com que o anuncio não voltasse para fila.

Apêndice

O manual para a interface Raspberry PI.

Manual do usuário

Operação para o cliente:

Serão utilizados botões, cada botão tem sua especificação ao seu lado.

- É possível escolher seu refrigerante através do seu botão, existe um botão para cada refrigerante, MEET e ETIRPS
- Ao apertar o botão adicionar moedas a maquina os botões que representam cada uma das moedas (R\$ 0,25, R\$ 0,50, R\$ 1,00) ficaram disponíveis, apertar o botão significa que esta sendo colocada a moeda correspondente na maquina.
- É possível ver os refrigerantes que podem ser vendidos na maquina através do botão listar produtos, essa opção apresenta os refrigerantes que estão cadastrados na maquina bem como sua disponibilidade.
- É possível ver os anúncios ao apertar o botão visualização de anúncios.

Operação do programador:

- Ao apertar o botão do modo programador, será solicitada uma senha, essa senha é 9090.
- É possível navegar no menu através do teclado, utilizando os números e Enter para confirmar.
- É possível cadastrar novos anunciantes na opção adicionar anunciante.
- É possível remover anúncios na opção remover anuncio.
- Todo anuncio cadastrado fica armazenado, para que o anuncio passe a aparecer no display é preciso carrega-lo, utilizando a opção carregar.
- É possível cadastrar novos produtos na maquina bem como sua disponibilidade, para isso acesse a opção adicionar produto.
- Para apagar produtos cadastrados, utilize a opção deletar produto.