Sistema de Mercado em Assembly 32bits

Gustavo Henrique Trassi Ganaza Yoshiyuki Fugie

14 de julho de 2025

Arquitetura do Sistema de Gerenciamento de Supermercado

- Sistema implementado em Assembly x86 32 bits
- Baseado em lista ligada para armazenamento de produtos
- Funcionalidades: CRUD, consultas financeiras, relatórios e persistência em disco

Arquivos e Estruturas

- **Arquivo principal:** supermercado.s (Assembly)
- **Armazenamento:** produtos.bin (binário)
- **Relatórios:** relatorio.txt (texto)
- **Instruções para compilação:** makefile
  - Compila com `gcc -m32`, portanto é necessário ter o pacote `gcc-multilib` instalado.

Armazenamento em Arquivo Binário

- **Arquivo:** produtos.bin (modo binário)
- **Processo de gravação (save_list):**
  - Abre arquivo com fopen em modo escrita ("wb")
  - Para cada nó:
    - Copia 148 bytes de dados (excluindo ponteiro) para um buffer
    - Escreve buffer no arquivo com fwrite
  - Fecha arquivo com fclose
- **Processo de leitura (load_list):**
  - Abre arquivo em modo leitura ("rb")
  - Lê blocos de 148 bytes (cada corresponde a um produto)
  - Para cada bloco: aloca memória, copia dados, insere na lista

Alocação de Espaço para Produtos

- Cada produto é um nó na lista ligada
- Tamanho total: 152 bytes (produto_size)
- Estrutura do nó:

```
; Offsets (em bytes):
;   0: Ponteiro para próximo nó (4 bytes)
;   4: Nome (50 bytes)
;  54: Lote (20 bytes)
;  74: Tipo (4 bytes - índice para tabela de tipos)
;  78: Dia/Mês/Ano (4 bytes cada)
;  90: Fornecedor (50 bytes)
; 140: Quantidade (4 bytes)
; 144: Preço de compra (4 bytes - centavos)
; 148: Preço de venda (4 bytes - centavos)
```

Menu de Operações

```
===== MENU =====
1. Adicionar produto
2. Buscar produto
3. Remover produto
4. Atualizar produto
5. Consultas Financeiras
6. Gerar relatório
7. Sair
Escolha:
```

Função add_product_interactive

Fluxo interativo via terminal:

1. Aloca memória com malloc(152)
2. Lê campos do usuário:
   - read_string_with_prompt para strings (nome, lote, fornecedor)
   - scanf para números (tipo, datas, quantidades, preços)
3. Insere nó na lista com insert_sorted (ordem alfabética por nome)

Exemplo add_product_interactive

```
Escolha: 1
Digite o nome do produto: Xícara
Digite o lote: 111
Tipos:
 01. Alimento
 02. Limpeza
 03. Utensílios
 04. Bebidas
 05. Frios
 06. Padaria
 07. Carnes
 08. Higiene
 09. Bebês
 10. Pet
 11. Congelados
 12. Hortifruti
 13. Eletronicos
 14. Vestuário
 15. Outros
Digite o tipo (1-15): 3
```

Exemplo add_product_interactive

```
Digite o tipo (1-15): 3
Digite o dia da validade: 1
Digite o mês da validade: 1
Digite o ano da validade: 2050
Digite o fornecedor: Starbucks
Digite a quantidade: 20
Digite o valor de compra (centavos): 2000
Digite o valor de venda (centavos): 5000
```

Função search_product

- **Entrada:** Nome do produto (via str_busca_prompt)
- **Algoritmo:**
  - Percorre a lista ligada
  - Para cada nó:
    - Compara nome com strcmp
    - Se igual, imprime detalhes com print_product
- **Saída:** Todos os produtos com nomes correspondentes

Função search_product

```
Escolha: 2
Digite o nome para buscar: Coca-Cola-2l
Nome: Coca-Cola-2l
Lote: 101
Tipo: Bebidas
Validade: 01/07/2026
Fornecedor: Coca-Cola
Quantidade: 300
Compra: 8.00
Venda: 12.00
---------------
Nome: Coca-Cola-2l
Lote: 100
Tipo: Bebidas
Validade: 01/01/2026
Fornecedor: Coca-Cola
Quantidade: 300
Compra: 8.00
Venda: 12.00
---------------
```

Função remove_product_interactive

- **Entrada:** Nome + lote do produto
- **Passos:**
  1. Busca nó na lista (compara nome e lote)
  2. Ajusta ponteiros:
     - Se nó anterior existe: anterior->next = atual->next
     - Se é o primeiro: head = atual->next
  3. Libera memória com free(atual)
- **Mensagens:** "Produto removido"ou "não encontrado"

Exemplo remove_product_interactive

```
===== MENU =====
1. Adicionar produto
2. Buscar produto
3. Remover produto
4. Atualizar produto
5. Consultas Financeiras
6. Gerar relatório
7. Sair
Escolha: 3
Digite o nome do produto a remover: Xícara
Digite o lote do produto a remover: 111
Produto removido com sucesso!
```

Função update_product_interactive

- **Entrada:** Nome + lote do produto
- **Fluxo:**
  1. Busca nó na lista
  2. Pergunta campo a atualizar (quantidade ou preço de venda)
  3. Lê novo valor e atualiza o campo no nó
- **Campos atualizáveis:**
  - Quantidade (offset 140)
  - Preço de venda (offset 148)

Exemplo update_product_interactive

```
Escolha: 4
Digite o nome do produto a atualizar: Picanha-kg
Digite o lote do produto a atualizar: 100
Qual campo deseja atualizar?
1. Quantidade
2. Valor de venda
Escolha: 2
Digite o novo valor de venda (centavos): 8000
Produto atualizado com sucesso!
```

Consultas Financeiras

Opções:

- **Total de compra:** Soma quantidade x preço_compra (via total_compra)
- **Total de venda:** Soma quantidade x preço_venda (via total_venda)
- **Lucro total:** total_venda - total_compra (via lucro_total)
- **Capital perdido:** Soma de produtos vencidos (compara datas via capital_perdido)

Formatação: Valores em reais/centavos (ex: printf("%d.%02d"))

Submenu de Consultas

```
===== CONSULTAS FINANCEIRAS =====
1. Total de compra
2. Total de venda
3. Lucro total
4. Capital perdido
5. Voltar
Escolha:
```

Função total_compra, total_venda e lucro_total

Total compra

- **Entrada:** Lista de produtos
- **Processo:**
  1. Percorre a lista ligada
  2. Para cada nó:
     - Multiplica quantidade pelo preço de compra/venda
     - Acumula o total

Total venda

- Similar ao total_compra, mas usa preço de venda

Lucro Total: chama total_venda e total_compra, e faz a subtração

Função capital_perdido

capital_perdido:

1. Pede a data atual ao usuário.
2. Para cada produto, compara a data de validade com a data atual usando `compare_dates`.
3. Se a validade for anterior à data atual, soma `quantidade * valor_compra` ao total perdido.

Exemplo total_compra

```
===== CONSULTAS FINANCEIRAS =====
1. Total de compra
2. Total de venda
3. Lucro total
4. Capital perdido
5. Voltar
Escolha: 1
Total gasto em compras: 131700.00
```

Exemplo total_venda

```
===== CONSULTAS FINANCEIRAS =====
1. Total de compra
2. Total de venda
3. Lucro total
4. Capital perdido
5. Voltar
Escolha: 2
Total estimado de vendas: 170200.00
```

Exemplo lucro_total

```
===== CONSULTAS FINANCEIRAS =====
1. Total de compra
2. Total de venda
3. Lucro total
4. Capital perdido
5. Voltar
Escolha: 3
Lucro estimado: 38500.00
```

Exemplo capital_perdido

```
===== CONSULTAS FINANCEIRAS =====
1. Total de compra
2. Total de venda
3. Lucro total
4. Capital perdido
5. Voltar
Escolha: 4
Digite o dia atual: 1
Digite o mês atual: 1
Digite o ano atual: 2100
Capital perdido: 131700.00
```

Função generate_report

- **Arquivo de saída:** relatorio.txt (modo texto)
- **Ordenação opcional:**
  - Por nome (padrão)
  - Por quantidade (crescente, via sort_by_quantity)
  - Por validade (mais antiga primeiro, via sort_by_date)
- **Técnica:**
  1. Converte lista em array de ponteiros
  2. Ordena array com bubble sort
  3. Grava produtos no arquivo com print_product_to_file (usa fprintf)

Submenu de Relatórios

```
Escolha: 6
Ordenar por:
1. Nome (padrão)
2. Quantidade em estoque
3. Data de validade (mais antiga primeiro)
Escolha:
```

Exemplo generate_report

```
----------------
Nome: Geladeira-375l
Lote: 100
Tipo: Eletronicos
Validade: 01/01/2050
Fornecedor: Brastemp
Quantidade: 20
Compra: 2600.00
Venda: 3300.00
----------------
Nome: Iphone-13-128gb
Lote: 100
Tipo: Eletronicos
Validade: 01/01/2050
Fornecedor: Apple
Quantidade: 20
Compra: 2000.00
Venda: 2500.00
----------------
Nome: Kit-Talher
Lote: 100
Tipo: Utensilios
Validade: 01/01/2050
Fornecedor: Tramontina
Quantidade: 50
Compra: 80.00
Venda: 100.00
----------------
```

Exemplo generate_report (Ordenação por Quantidade)

```
Nome: Picanha-kg
Lote: 100
Tipo: Carnes
Validade: 01/08/2025
Fornecedor: Swift
Quantidade: 100
Compra: 65.00
Venda: 80.00
---------------
Nome: Kit-Talher
Lote: 100
Tipo: Utensilios
Validade: 01/01/2050
Fornecedor: Tramontina
Quantidade: 50
Compra: 80.00
Venda: 100.00
---------------
Nome: Geladeira-375l
Lote: 100
Tipo: Eletronicos
Validade: 01/01/2050
Fornecedor: Brastemp
Quantidade: 20
Compra: 2600.00
Venda: 3300.00
---------------
```

Exemplo generate_report (Ordenação por Validade)

```
Nome: Picanha-kg
Lote: 100
Tipo: Carnes
Validade: 01/08/2025
Fornecedor: Swift
Quantidade: 100
Compra: 65.00
Venda: 80.00
---------------
Nome: Arroz-Japones-5kg
Lote: 100
Tipo: Alimento
Validade: 01/01/2026
Fornecedor: Guin
Quantidade: 100
Compra: 80.00
Venda: 110.00
---------------
Nome: Coca-Cola-2l
Lote: 100
Tipo: Bebidas
Validade: 01/01/2026
Fornecedor: Coca-Cola
Quantidade: 300
Compra: 8.00
Venda: 12.00
```