

roteiro-02/ex01-01.c

```
1  #include <stdio.h>
2  #include <stdlib.h>
3  #include <string.h>
4
5  #define NOME_CHARS 50
6  #define CONTAS_MAX 100
7
8  typedef struct {
9      int numero;
10     char titular[NOME_CHARS + 1];
11     double saldo;
12 } ContaBancaria;
13
14 typedef struct {
15     ContaBancaria contas[CONTAS_MAX];
16     int quantidade;
17 } Banco;
18
19 enum OPCOES {
20     SAIR = 0,
21     CRIAR,
22     DEPOSITAR,
23     SACAR,
24     CONSULTAR,
25     IMPRIMIR
26 };
27
28 Banco criarBanco() {
29     Banco novo;
30     novo.quantidade = 0;
31     return novo;
32 }
33
34 void criarConta(ContaBancaria* c, int numero, char* titular) {
35     c->numero = numero;
36     strcpy(c->titular, titular);
37     c->saldo = 0.0;
38 }
39
40 void depositar(ContaBancaria* c, double valor) {
41     c->saldo += valor;
42 }
43
44 void sacar(ContaBancaria* c, double valor) {
45     if (c->saldo >= valor) {
46         c->saldo -= valor;
47     }
48 }
```

```

49
50 double consultarSaldo(ContaBancaria* c) {
51     return c->saldo;
52 }
53
54 void imprimirInfo(ContaBancaria* c) {
55     printf("Numero: %d\n", c->numero);
56     printf("Titular: %s\n", c->titular);
57     printf("Saldo: %.2lf\n", c->saldo);
58 }
59
60 int getOpcao() {
61     int opcao;
62
63     printf("\n=====\\n");
64     printf("(%) Criar conta\\n", CRIAR);
65     printf("(%) Depositar\\n", DEPOSITAR);
66     printf("(%) Sacar\\n", SACAR);
67     printf("(%) Consultar saldo\\n", CONSULTAR);
68     printf("(%) Imprimir informacoes\\n", IMPRIMIR);
69     printf("(%) Sair\\n", SAIR);
70     printf("=====\\n");
71
72     printf("Opcao escolhida: ");
73     scanf("%d", &opcao);
74
75     printf("\\n");
76     return opcao;
77 }
78
79 ContaBancaria* getConta(int numero, Banco* banco) {
80     for (int i = 0; i < banco->quantidade; i++) {
81         if (banco->contas[i].numero == numero) {
82             return &banco->contas[i];
83         }
84     }
85     return NULL;
86 }
87
88 int main() {
89     Banco banco = criarBanco();
90     int numero;
91     double valor;
92     char titular[NOME_CHARS + 1];
93     ContaBancaria* conta;
94
95     int sair = 0;
96     while (!sair) {
97         switch (getOpcao()) {
98             case SAIR:
99                 sair = 1;
100                 break;

```

```

101         case CRIAR:
102             printf("Numero: ");
103             scanf("%d", &numero);
104             setbuf(stdin, NULL);
105             printf("Titular: ");
106             fgets(titular, NOME_CHARS + 1, stdin);
107             if (titular[strlen(titular) - 1] == '\n')
108                 titular[strlen(titular) - 1] = '\0';
109             setbuf(stdin, NULL);
110
111             conta = getConta(numero, &banco);
112             if (conta == NULL) {
113                 criarConta(&banco.contas[banco.quantidade], numero,
titular);
114                 banco.quantidade++;
115                 printf("Conta criada com sucesso\n");
116             } else {
117                 printf("Conta ja existente\n");
118             }
119             break;
120         case DEPOSITAR:
121             printf("Numero: ");
122             scanf("%d", &numero);
123             printf("Valor: ");
124             scanf("%lf", &valor);
125
126             conta = getConta(numero, &banco);
127             if (conta != NULL) {
128                 depositar(conta, valor);
129                 printf("Deposito efetuado com sucesso\n");
130             } else {
131                 printf("Conta nao encontrada\n");
132             }
133             break;
134         case SACAR:
135             printf("Numero: ");
136             scanf("%d", &numero);
137             printf("Valor: ");
138             scanf("%lf", &valor);
139
140             conta = getConta(numero, &banco);
141             if (conta != NULL) {
142                 sacar(conta, valor);
143                 printf("Saque efetuado com sucesso\n");
144             } else {
145                 printf("Conta nao encontrada\n");
146             }
147             break;
148         case CONSULTAR:
149             printf("Numero: ");
150             scanf("%d", &numero);
151
152             conta = getConta(numero, &banco);

```

```
153         if (conta != NULL) {
154             printf("Saldo = %2lf\n", consultarSaldo(conta));
155         } else {
156             printf("Conta nao encontrada\n");
157         }
158         break;
159     case IMPRIMIR:
160         printf("Numero: ");
161         scanf("%d", &numero);
162
163         conta = getConta(numero, &banco);
164         if (conta != NULL) {
165             imprimirInfo(conta);
166         } else {
167             printf("Conta nao encontrada\n");
168         }
169         break;
170     default:
171         printf("Opcao invalida - Tente novamente\n");
172         break;
173     }
174 }
175
176 return 0;
177 }
```



gabriel-dp@gabriel-dp: ~/Desktop/dev/c/lab2/roteiro-02



gabriel-dp@gabriel-dp:~/Desktop/dev/c/lab2/roteiro-02\$./ex01-01

```
=====
(1) Criar conta
(2) Depositar
(3) Sacar
(4) Consultar saldo
(5) Imprimir informacoes
(0) Sair
=====
```

Opcao escolhida: 1

```
Numero: 23
Titular: Gabriel de Paula
Conta criada com sucesso
```

```
=====
(1) Criar conta
(2) Depositar
(3) Sacar
(4) Consultar saldo
(5) Imprimir informacoes
(0) Sair
=====
```

Opcao escolhida: 2

```
Numero: 23
Valor: 100.50
Deposito efetuado com sucesso
```

```
=====
(1) Criar conta
(2) Depositar
(3) Sacar
(4) Consultar saldo
(5) Imprimir informacoes
(0) Sair
=====
```

Opcao escolhida: 3

```
Numero: 23
Valor: 1.50
Saque efetuado com sucesso
```



Numero: 23
Valor: 1.50
Saque efetuado com sucesso

=====
(1) Criar conta
(2) Depositar
(3) Sacar
(4) Consultar saldo
(5) Imprimir informacoes
(0) Sair

=====
Opcao escolhida: 4

Numero: 23
Saldo = 99.000000

=====
(1) Criar conta
(2) Depositar
(3) Sacar
(4) Consultar saldo
(5) Imprimir informacoes
(0) Sair

=====
Opcao escolhida: 5

Numero: 23
Numero: 23
Titular: Gabriel de Paula
Saldo: 99.00

=====
(1) Criar conta
(2) Depositar
(3) Sacar
(4) Consultar saldo
(5) Imprimir informacoes
(0) Sair

=====
Opcao escolhida: 0

gabriel-dp@gabriel-dp:~/Desktop/dev/c/lab2/roteiro-02\$

roteiro-02/ex01-02.c

```
1  #include <stdio.h>
2  #include <stdlib.h>
3  #include <string.h>
4
5  #define NOME_CHARS 50
6
7  typedef struct {
8      char nome[NOME_CHARS];
9      double preco;
10     int quantidade;
11 } Produto;
12
13 typedef struct {
14     Produto produtos[100];
15     int produtos_total;
16 } CatalogoProdutos;
17
18 enum OPCOES {
19     SAIR = 0,
20     ADICIONAR,
21     VERIFICAR,
22     IMPRIMIR
23 };
24
25 void criarCatalogo(CatalogoProdutos *c) {
26     c->produtos_total = 0;
27 }
28
29 void adicionarProduto(CatalogoProdutos *c, char *nome, double preco, int
quantidade) {
30     Produto novo;
31     strcpy(novo.nome, nome);
32     novo.preco = preco;
33     novo.quantidade = quantidade;
34
35     c->produtos[c->produtos_total] = novo;
36 }
37
38 int verificarEstoque(CatalogoProdutos *c, char *nome) {
39     for (int i = 0; i < c->produtos_total; i++) {
40         if (strcmp(nome, c->produtos[i].nome) == 0) {
41             return c->produtos[i].quantidade;
42         }
43     }
44
45     return 0;
46 }
47
48 void imprimirCatalogo(CatalogoProdutos *c) {
49     for (int i = 0; i < c->produtos_total; i++) {
50         printf("Produto %d { Nome = %s | Preco = %.2lf | Quantidade = %d }\n",
i + 1, c->produtos[i].nome, c->produtos[i].preco, c->produtos[i].quantidade);
51     }
52 }
53
```

```

54 int getOpcao() {
55     int opcao;
56
57     printf("\n=====\\n");
58     printf("(%) Adicionar produto\\n", ADICIONAR);
59     printf("(%) Verificar estoque\\n", VERIFICAR);
60     printf("(%) Imprimir catalogo\\n", IMPRIMIR);
61     printf("(%) Sair\\n", SAIR);
62     printf("=====\\n");
63     printf("Opcao escolhida: ");
64     scanf("%d", &opcao);
65
66     printf("\\n");
67     return opcao;
68 }
69
70 Produto *getProduto(CatalogoProdutos *catalogo, char *nome) {
71     for (int i = 0; i < catalogo->produtos_total; i++) {
72         if (strcmp(catalogo->produtos[i].nome, nome) == 0) {
73             return &catalogo->produtos[i];
74         }
75     }
76     return NULL;
77 }
78
79 int main() {
80     CatalogoProdutos catalogo;
81     criarCatalogo(&catalogo);
82
83     int quantidade;
84     double preco;
85     char nome[NOME_CHARS + 1];
86
87     int sair = 0;
88     while (!sair) {
89         switch (getOpcao()) {
90             case SAIR:
91                 sair = 1;
92                 break;
93             case ADICIONAR:
94                 setbuf(stdin, NULL);
95                 printf("Nome: ");
96                 fgets(nome, NOME_CHARS + 1, stdin);
97                 if (nome[strlen(nome) - 1] == '\\n')
98                     nome[strlen(nome) - 1] = '\\0';
99                 setbuf(stdin, NULL);
100                 printf("Preco: ");
101                 scanf("%lf", &preco);
102                 printf("Quantidade: ");
103                 scanf("%d", &quantidade);
104
105                 if (getProduto(&catalogo, nome) == NULL) {
106                     adicionarProduto(&catalogo, nome, preco, quantidade);
107                     catalogo.produtos_total++;
108                     printf("Produto adicionado com sucesso\\n");
109                 } else {
110                     printf("Produto ja existente\\n");
111                 }
112                 break;

```



```

113         case VERIFICAR:
114             setbuf(stdin, NULL);
115             printf("Nome: ");
116             fgets(nome, NOME_CHARS + 1, stdin);
117             if (nome[strlen(nome) - 1] == '\n')
118                 nome[strlen(nome) - 1] = '\0';
119             setbuf(stdin, NULL);
120
121             if (getProduto(&catalogo, nome) != NULL) {
122                 printf("Quantidade = %d\n", verificarEstoque(&catalogo,
nome));
123             } else {
124                 printf("Produto nao encontrado\n");
125             }
126             break;
127         case IMPRIMIR:
128             imprimirCatalogo(&catalogo);
129             break;
130         default:
131             printf("Opcao invalida - Tente novamente\n");
132             break;
133     }
134 }
135
136 return 0;
137 }

```



gabriel-dp@gabriel-dp: ~/Desktop/dev/c/lab2/roteiro-02



gabriel-dp@gabriel-dp:~/Desktop/dev/c/lab2/roteiro-02\$./ex01-02

=====

- (1) Adicionar produto
- (2) Verificar estoque
- (3) Imprimir catalogo
- (0) Sair

=====

Opcao escolhida: 1

Nome: Sorvete

Preco: 24.50

Quantidade: 23

Produto adicionado com sucesso

=====

- (1) Adicionar produto
- (2) Verificar estoque
- (3) Imprimir catalogo
- (0) Sair

=====

Opcao escolhida: 1

Nome: Cereal

Preco: 12.99

Quantidade: 57

Produto adicionado com sucesso

=====

- (1) Adicionar produto
- (2) Verificar estoque



gabriel-dp@gabriel-dp: ~/Desktop/dev/c/lab2/roteiro-02



=====

(1) Adicionar produto
(2) Verificar estoque
(3) Imprimir catalogo
(0) Sair

=====

Opcao escolhida: 2

Nome: Sorvete

Quantidade = 23

=====

(1) Adicionar produto
(2) Verificar estoque
(3) Imprimir catalogo
(0) Sair

=====

Opcao escolhida: 3

Produto 1 { Nome = Sorvete | Preco = 24.50 | Quantidade = 23 }

Produto 2 { Nome = Cereal | Preco = 12.99 | Quantidade = 57 }

=====

(1) Adicionar produto
(2) Verificar estoque
(3) Imprimir catalogo
(0) Sair

=====

Opcao escolhida: 0

gabriel-dp@gabriel-dp:~/Desktop/dev/c/lab2/roteiro-02\$

2) Análise de Complexidade

2.1

$$8n^2 > 64n \log n$$

$$n > 8 \log n$$

$$\frac{n}{8} > \log_2 n$$

$$[44, \infty[//$$

2.2

$$100n^2 < 2^n$$

$$\log_2(100n^2) < n$$

$$\log_2(100) + 2 \log_2(n) < n$$

$$6.6 + 2 \log_2(n) < n$$

$$15 //$$

2.3

A notação " O " define que o limite superior de $g(n)$ é $f(n)$.

2.4

A notação " Ω " define que o limite inferior de $g(n)$ é $f(n)$.

2.5

Não é possível que o mínimo seja $O(n^2)$, já que a notação define o limite superior (máximo).

2.6

$$a(n) = n^2 - n + 500$$

$$b(n) = 47n + 47$$

$$n^2 - n + 500 < 47n + 47$$

$$n^2 - 48n + 453 < 0$$

$$12,9 < n < 35,1 //$$

$$[13, 35]$$

$$\Delta = \sqrt{48^2 - 4 \cdot 453}$$

$$\Delta \approx 22,2$$

$$\frac{48 \pm 22,2}{2} = 24 \pm 11,1$$

2.7

$$\sum_{j=1}^{n-1} \sum_{k=1}^j k$$

$$O(n^3)$$

$$\frac{n^3 - 3n^2 + 2n}{3}$$

2.8

$n-1$ passos para

encontrar o maior. Se é $\Omega(n)$,
logo é Θ .

$C=1$ em $\Omega(n)$ e $C=2$ em $O(n)$