

RUSLEY SILVA SANTOS

LUCAS BASTOS

RÔMULO CAXIAS

ANDRE DANILO

CONTROLE DE ESTOQUE

RUSLEY SILVA SANTOS

LUCAS BASTOS

RÔMULO

ANDRE DANILO

CONTROLE DE ESTOQUE

Trabalho apresentado à Universidade de Vila Velha na disciplina de Programação de Computados I e Sistemas Operacionais do curso Bacharel em Sistemas de Informação.

Orientador:

Vinicius Rosalen da Silva

**SUMÁRIO**

**1 I**[**NTRODUÇÃO**](https://docs.google.com/document/d/1hI1R5nHk6Lv_wZ8KuJZiv09_ETB0fYIIcu0D-Gn5wFM/edit#heading=h.gjdgxs)\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_4

**2** [**DESENVOLVIMENTO**](https://docs.google.com/document/d/1hI1R5nHk6Lv_wZ8KuJZiv09_ETB0fYIIcu0D-Gn5wFM/edit#heading=h.nnkcjjol0ztc)\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_5

**4** [**CONCLUSÃO**](https://docs.google.com/document/d/1hI1R5nHk6Lv_wZ8KuJZiv09_ETB0fYIIcu0D-Gn5wFM/edit#heading=h.6clnw93zwwdh)\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_11

**5** [**REFERÊNCIAS**](https://docs.google.com/document/d/1hI1R5nHk6Lv_wZ8KuJZiv09_ETB0fYIIcu0D-Gn5wFM/edit#heading=h.jlc9qzm5r49y)\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_12

**1 – MINI MUNDO**

Uma empresa fará o controle do seu estoque usando um software desenvolvido por você, especialmente para este fim. Ela trabalha com no máximo 100 tipos de produtos, conhecendo para cada um deles: código, nome, quantidade em estoque, preço unitário de custo, quantidade vendida e o preço unitário de venda. Como sugestão, pelo menos cinco vetores podem ser usados para o armazenamento dos dados dos produtos: um com os códigos (valores do tipo int), outro com os preços unitários de custo (valores do tipo float), outro com os preços unitários de venda (valores do tipo float), um quarto com as quantidades em estoque (valores do tipo int) e um quinto com as quantidades vendidas (valores do tipo int). Use um vetor de string para armazenar os nomes dos produtos.

* 1. FUNCIONALIDADES
* Inclusão de cadastros. Ao cadastrar, o programa deve garantir que não ocorram dois produtos com o mesmo código, que o preço seja um valor maior que zero e que a quantidade seja maior ou igual a zero;
* Alteração de um cadastro específico. O cadastro a ser alterado deve ser identificado pelo código. Somente o preço e a quantidade podem ser alterados. Utilize o Método de Pesquisa Sequencial para localizar um código na lista de produtos;
* Exclusão de um cadastro específico. O cadastro a ser excluído deve ser identificado pelo código. Utilize o Método de Pesquisa Sequencial para localizar um código na lista de produtos;
* Relatório contendo todos os produtos cadastrados, em ordem crescente de código. Utilize o Método da Bolha para ordenar a lista de produtos;
* Relatório com os dados de um produto específico, identificado pelo código. Utilize o Método de Pesquisa para localizar um código na lista de produtos;
* Relatório com todos os dados do produto e o cálculo do valor total de um determinado produto, o valor total de venda e o lucro obtido.

1. **DESENVOLVIMENTO**

O sistema foi desenvolvido em C utilizando funções em alguns casos a fim de reaproveitar o código e deixa-lo mais limpo e legível.

* 1. CÓDIGO

Confira abaixo todo o código estruturado do sistema com comentários explicando cada função.

|  |
| --- |
|  |
| /\* |
|  | \*Algoritimo desenvolvido para o curso de Bacharel em Sistemas de Informação(UVV) por Rusley S. Santos |
|  | \*/ |
|  | #include <stdio.h> |
|  | #include <stdlib.h> |
|  | #include <locale.h> |
|  | #include <stdbool.h> |
|  | #define TAM 101  //Declaração de variáveis |
|  |  |
|  | int Codigo[TAM], Estoque[TAM], Vendido[TAM],i,j,Opcao, tes=1, Aux=0,AuxMenu, flagCad=0,flagBusca=0, AuxBusca, Count,x,y,AuxInt; |
|  | float PrecoCusto[TAM], PrecoVenda[TAM], TotalDinheiroCompra,TotalDinheiroVenda,Lucro,AuxFloat; |
|  | //char Nome[TAM][100];  /\*  \*Foi utilizado ponteiro nos tipos char devido problemas para apagar ou atribuir novos valores nos \*vetores.  \*/ |
|  | char \*Nome[TAM],\*AuxName=NULL; |
|  |  |
|  | int main() |
|  | {  //Seleciona a linguagem padrão da aplicação. |
|  | setlocale(LC\_ALL, "Portuguese");  //Chama o menu principal da aplicação. |
|  | PrincipalMenu(); |
|  | } |
|  | //Verifica se o ID e igual a algum já cadastrado |
|  | bool Valor(int Codigo1) |
|  | { |
|  | bool flag = false; |
|  |  |
|  | for(j=0;j<TAM;j++) |
|  | { |
|  | if(Codigo[j] == Codigo1) |
|  | { |
|  | flag = true; |
|  | printf("O código %i já foi utilizado, digite outro! \n", Codigo[j]); |
|  | } |
|  | } |
|  | return flag; |
|  | }  //Esta função recebe como parâmetro a opção digitada pelo usuário. |
|  |  |
|  | int Logica(int Opcao){ |
|  | switch(Opcao) |
|  | {  /\*  \*Caso o usuário escolha a opção 1 ele irá realizar o cadastro do produto com seus respectivos \*atributos.  \*/ |
|  | case 1: |
|  | //Cadatrar |
|  | system("cls"); |
|  | for(i=0;i<TAM;i++) |
|  | { |
|  | do{ |
|  | printf("Digite o código do produto: \n"); |
|  | scanf("%i",&Aux); |
|  | } |
|  | while(Valor(Aux)); |
|  | Codigo[i]=Aux; |
|  |  |
|  | fflush(stdin); |
|  | printf("Digite o nome do produto: \n"); |
|  | //scanf("%[^\n]s", Nome[i]); |
|  | scanf("%[^\n]s",&Nome[i]); |
|  | printf("\n%s\n",&Nome[i]); |
|  | fflush(stdin); |
|  |  |
|  | printf("Digite a quantidade em estoque: \n"); |
|  | scanf("%i",&Estoque[i]); |
|  |  |
|  | printf("Digite a quantidade vendida: \n"); |
|  | scanf("%i",&Vendido[i]); |
|  |  |
|  | printf("Digite o preço de venda: \n"); |
|  | scanf("%f",&PrecoVenda[i]); |
|  | fflush(stdin); |
|  |  |
|  | printf("Digite o preço de custo: \n"); |
|  | scanf("%f",&PrecoCusto[i]); |
|  | fflush(stdin); |
|  | printf("------------------------------\n"); |
|  | }  //Informa que existem produtos cadastrados no sistema. |
|  | flagCad = 1;  /\*  \*Pergunta se o usuário deseja retornar ao menu, note que esse trecho e repetido  \*diversas vezes no código, função e a mesma enviar como parâmetro para a função voltar \*ao menu a opção escolhida pelo usuário se ele deseja ou não voltar ao menu principal.  \*/ |
|  | printf("Todos os produtos foram cadastrados com sucesso!\n"); |
|  | printf("Deseja voltar ao menu principal?(1 = sim / 2 = não) \n"); |
|  | scanf("%i",&AuxMenu); |
|  | VoltarMenu(AuxMenu); |
|  | break; |
|  |  |
|  | case 2: |
|  | //Alterar  /\*  \*Caso o usuário deseje alterar algum produto ele pode digitar o ID do produto em seguida \*sobrescrever os valores anteriormente digitados.  \*/ |
|  | system("cls"); |
|  |  |
|  | printf("Digite o ID do produto que deseja alterar:\n"); |
|  | scanf("%i",&AuxBusca); |
|  |  |
|  | for(i=0;i<TAM;i++) |
|  | { |
|  | if(Codigo[i]==AuxBusca) |
|  | { |
|  | printf("Digite a quantidade em estoque: \n"); |
|  | scanf("%i",&Estoque[i]); |
|  |  |
|  | printf("Digite o preço de venda: \n"); |
|  | scanf("%f",&PrecoVenda[i]); |
|  | fflush(stdin); |
|  |  |
|  | flagBusca = 1; |
|  |  |
|  | printf("Produto alterado com sucesso!!!.\n\n"); |
|  | printf("Deseja voltar ao menu principal?(1 = sim / 2 = não) \n"); |
|  | scanf("%i",&AuxMenu); |
|  | VoltarMenu(AuxMenu); |
|  | } |
|  | } |
|  | if(flagBusca==0) |
|  | { |
|  | printf("Não foi encontrato nenhum produto com esse ID.\n\n"); |
|  | printf("Deseja voltar ao menu principal?(1 = sim / 2 = não) \n"); |
|  | scanf("%i",&AuxMenu); |
|  | VoltarMenu(AuxMenu); |
|  | } |
|  | break; |
|  |  |
|  | case 3: |
|  | //Excluir  /\*  \*Em conversa com o professor foi permitido que ao apagar os valores fosse aceito que \*eles recebessem seu valor igual a 0, nesta função o usuário digita o ID e todos os \*valores naquela posição em todos os vetores da aplicação são zerados.  \*/ |
|  | system("cls"); |
|  |  |
|  | printf("Digite o ID do produto que deseja excluir:\n"); |
|  | scanf("%i",&AuxBusca); |
|  |  |
|  | for(i=0;i<TAM;i++) |
|  | { |
|  | if(Codigo[i]==AuxBusca) |
|  | { |
|  | Codigo[i] = 0; |
|  | Nome[i] = '0'; |
|  | Vendido[i] = 0; |
|  | Estoque[i] = 0; |
|  | PrecoVenda[i] = 0; |
|  | PrecoCusto[i] = 0; |
|  |  |
|  | flagBusca = 1; |
|  | printf("\n"); |
|  | printf("Produto excluido com sucesso.\n\n"); |
|  | printf("Deseja voltar ao menu principal?(1 = sim / 2 = não) \n"); |
|  | scanf("%i",&AuxMenu); |
|  | VoltarMenu(AuxMenu); |
|  | } |
|  | } |
|  | if(flagBusca==0) |
|  | { |
|  | printf("Não foi encontrato nenhum produto com esse ID.\n\n"); |
|  | printf("Deseja voltar ao menu principal?(1 = sim / 2 = não) \n"); |
|  | scanf("%i",&AuxMenu); |
|  | VoltarMenu(AuxMenu); |
|  | } |
|  |  |
|  | case 4: |
|  | //Caso o usuário escolha a opção 4 todos os valores são ordenados do ID menor para o maior. |
|  | system("cls"); |
|  | for(x=0;x<TAM;x++) |
|  | { |
|  | for(y=0;y<TAM-1;y++) |
|  | { |
|  | if(Codigo[y]>Codigo[y+1]) |
|  | { |
|  | AuxInt = Codigo[y]; |
|  | Codigo[y] = Codigo[y+1]; |
|  | Codigo[y+1] = AuxInt; |
|  |  |
|  | AuxName=Nome[y]; |
|  | Nome[y] = Nome[y+1]; |
|  | Nome[y+1] = AuxName; |
|  |  |
|  | AuxInt = Estoque[y]; |
|  | Estoque[y] = Estoque[y+1]; |
|  | Estoque[y+1] = AuxInt; |
|  |  |
|  | AuxInt = Vendido[y]; |
|  | Vendido[y] = Vendido[y+1]; |
|  | Vendido[y+1] = AuxInt; |
|  |  |
|  | AuxFloat = PrecoVenda[y]; |
|  | PrecoVenda[y] = PrecoVenda[y+1]; |
|  | PrecoVenda[y+1] = AuxFloat; |
|  |  |
|  | AuxFloat = PrecoCusto[y]; |
|  | PrecoCusto[y] = PrecoCusto[y+1]; |
|  | PrecoCusto[y+1] = AuxFloat; |
|  | } |
|  | } |
|  | } |
|  |  |
|  | if(flagCad != 0) |
|  | { |
|  | /\*  \* Após os valores serem ordenados todos são exibidos.  \* Exemplo: se o usuário inserir os seguintes códigos 3,5 e 4 eles serão ordenados \* para 3,4 e 5.  \*/ |
|  | for(i=0;i<TAM;i++) |
|  | { |
|  | printf("ID: %i\n",Codigo[i]); |
|  | printf("Nome: %s\n",&Nome[i]); |
|  | printf("Estoque: %i\n",Estoque[i]); |
|  | printf("Vendido: %i\n",Vendido[i]); |
|  | printf("Preço de custo: %.2f\n",PrecoCusto[i]); |
|  | printf("Preco de Venda: %.2f\n",PrecoVenda[i]); |
|  | //Esse trecho calcula o lucro da loja. |
|  | TotalDinheiroCompra=Vendido[i]\*PrecoCusto[i]; |
|  | TotalDinheiroVenda=Vendido[i]\*PrecoVenda[i]; |
|  | Lucro = TotalDinheiroVenda-TotalDinheiroCompra; |
|  | printf("Lucro: %.2f\n",Lucro); |
|  | printf("------------------------------\n"); |
|  | } |
|  |  |
|  | printf(""); |
|  | printf("Deseja voltar ao menu principal?(1 = sim / 2 = não) \n"); |
|  | scanf("%i",&AuxMenu); |
|  | VoltarMenu(AuxMenu); |
|  | } |
|  | else |
|  | { |
|  | printf("Você precisa cadastrar produtos!!!\n\n"); |
|  | printf("Deseja voltar ao menu principal?(1 = sim / 2 = não) \n"); |
|  | scanf("%i",&AuxMenu); |
|  | VoltarMenu(AuxMenu); |
|  | } |
|  |  |
|  | case 5: |
|  | //Consultar por ID  //Caso o usuário escolha a opção 5 e feito uma busca simples por ID. |
|  | system("cls"); |
|  | printf("Digite o código do produto\n"); |
|  | scanf("%i",&AuxBusca); |
|  | for(i=0;i<TAM;i++) |
|  | { |
|  | if(Codigo[i]==AuxBusca) |
|  | { |
|  | printf("------------------------------\n"); |
|  | printf("ID: %i\n",Codigo[i]); |
|  | printf("Nome: %s\n",&Nome[i]); |
|  | //printf("Nome: %s\n",Nome[i]); |
|  | printf("Estoque: %i\n",Estoque[i]); |
|  | printf("Vendido: %i\n",Vendido[i]); |
|  | printf("Preço de custo: %.2f\n",PrecoCusto[i]); |
|  | printf("Preco de Venda: %.2f\n",PrecoVenda[i]); |
|  |  |
|  | TotalDinheiroCompra=Vendido[i]\*PrecoCusto[i]; |
|  | TotalDinheiroVenda=Vendido[i]\*PrecoVenda[i]; |
|  | Lucro = TotalDinheiroVenda-TotalDinheiroCompra; |
|  | printf("Lucro: %.2f\n",Lucro); |
|  | printf("------------------------------\n"); |
|  | flagBusca = 1; |
|  |  |
|  | printf("Deseja voltar ao menu principal?(1 = sim / 2 = não) \n"); |
|  | scanf("%i",&AuxMenu); |
|  | VoltarMenu(AuxMenu); |
|  | } |
|  | } |
|  | if(flagBusca==0) |
|  | { |
|  | printf("Não foi encontrato nenhum produto com esse ID.\n\n"); |
|  | printf("Deseja voltar ao menu principal?(1 = sim / 2 = não) \n"); |
|  | scanf("%i",&AuxMenu); |
|  | VoltarMenu(AuxMenu); |
|  | } |
|  | break; |
|  | case 6: |
|  | exit(0); |
|  | break; |
|  | default: |
|  | exit(0); |
|  | break; |
|  | } |
|  | } |
|  |  |
|  | //Função responsável por iniciar o menu. |
|  | int PrincipalMenu(){ |
|  |  |
|  | printf("---------------->CAD PRODUTO<---------------- \n"); |
|  | printf("1 - Cadastrar produtos \n"); |
|  | printf("2 - Alterar produto \n"); |
|  | printf("3 - Excluir produto \n"); |
|  | printf("---------------->RELATÓRIOS<-----------------\n"); |
|  | printf("4 - Consultar todos os produtos\n"); |
|  | printf("5 - Consultar por ID\n\n"); |
|  | printf("6 - Sair \n"); |
|  | scanf("%i",&Opcao); |
|  | Logica(Opcao); |
|  | return Opcao; |
|  | } |
|  |  |
|  | int VoltarMenu(int AuxMenu){ |
|  |  |
|  | switch(AuxMenu) |
|  | { |
|  | case 1: |
|  | system("cls"); |
|  | PrincipalMenu(); |
|  | break; |
|  | case 2: |
|  | exit(0); |
|  | break; |
|  | default: |
|  | exit(0); |
|  | break; |
|  |  |
|  | } |
|  | } |
|  |  |

Para melhor compreensão e trabalho em grupo o código foi adicionado a um repositório no GitHub https://github.com/rusleysantos/estoquecontrolefaculdade também e recomendado que seja baixado e executado para fins de teste por ele.

**CONCLUSÃO**

Com o desenvolvimento do sistema foi possível aprender novas técnicas de programação e boas práticas para criar código em uma linguagem estruturada, em destaque o aprendizado de ponteiros que facilitam o trabalho de conjuntos do tipo *char* e a ordenação de valores através do método *Bubble Sort.*

**REFERÊNCIAS**

LINGUAGEM C. Ponteiros em C. 2018. Disponível em: < http://linguagemc.com.br/ponteiros-em-c/>. Acesso em: 11 Jun. 2018.

CÓDIGO FONTE. Método Bolha. 2018. Disponível em: < https://www.codigofonte.com.br/codigos/metodo-bolha-bubble-sort>. Acesso em: 11 Jun. 2018.