CAPÍTULO

13 Desafios

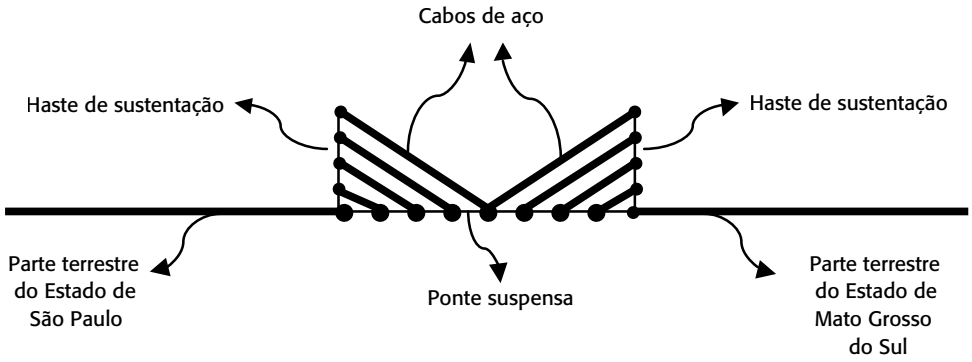
13.1 Desafio 1

Segundo o dicionário, a palavra PONTE é definida na ARQUITETURA como uma construção sólida em betão, aço ou madeira, destinada a estabelecer comunicação entre dois pontos separados por um curso de água ou por uma depressão de terreno.

A atual presidente da República vai beneficiar alguns estados do BRASIL com a construção de obras de grande magnitude e nosso estado será agraciado com uma ponte suspensa que será considerada um marco na construção civil, que ligará o estado de Mato Grosso do Sul ao estado de São Paulo.

Veja a foto a seguir com um exemplo de ponte suspensa já existente em Porto Alegre (RS) e que se difere da obra a ser construída, pois na ponte ilustrada, o material usado na sustentação foi uma corda especial, e isso impossibilita a passagem de veículos pesados. Assim, como o tráfego previsto para a ponte a ser construída envolve todo tipo de veículo, serão utilizados cabos de aço importados da ALEMANHA.

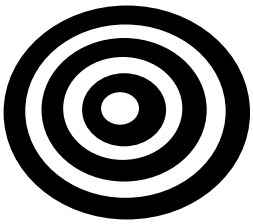
A seguir, apresentamos o projeto da ponte a ser construída.



O comprimento da ponte suspensa será fornecido pelo usuário em quilômetros e deve estar entre 2 e 4. Sabe-se que a ponte terá quatro hastes de sustentação, duas na parte terrestre do estado de Mato Grosso do Sul e outras duas na parte terrestre do estado de São Paulo, e estas ficam nas margens direita e esquerda. Tem-se, ainda, que a altura da haste, e onde deve ser fixado o cabo de aço mais alto, deve ser de $^{1}/_{20}$ da extensão da ponte. O cabo de aço mais alto será fixado obrigatoriamente no meio da ponte. Cada haste, num total de quatro, sustentará cinco cabos de aço equidistantes.

Faça um programa para calcular quantos metros de cabos de aço serão necessários para construir a ponte suspensa.

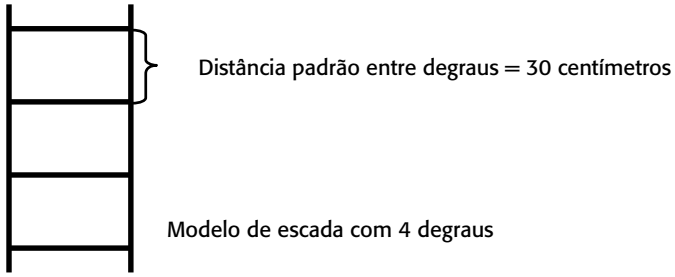
13.2 Desafio 2



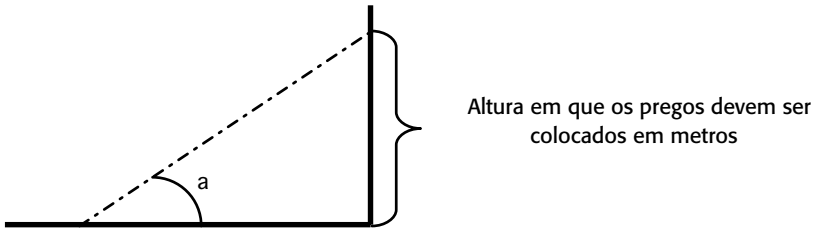
Uma fábrica de brinquedos precisa produzir alvos feitos de papelão. Para tal produção, será necessário adquirir o material, ou seja, o papelão. Assim, a fábrica solicitou à equipe de desenvolvimento que fizesse um programa para calcular a quantidade de papelão a ser comprada para a confecção de 5 mil alvos. Sabe-se que o brinquedo terá seis círculos sobrepostos e que todos os círculos apresentam a mesma distância do círculo imediatamente menor. O diâmetro do círculo maior será dado pelo usuário.

13.3 Desafio 3

Uma pessoa deseja colocar pregos a determinada altura do chão. Para isso, deverá comprar uma escada e apoiá-la na parede. Essa escada formará um ângulo conhecido como chão. Pelas especificações do Inmetro, todas as escadas produzidas no Brasil têm uma distância de 30 centímetros entre os degraus. Assim, deve-se construir um programa para saber que tipo de escada comprar, ou seja, quantos degraus deve ter a escada para que a altura do prego seja atingida, mesmo que aproximadamente. Desconsidere a altura da pessoa que fará o serviço.



Escada aprovada pelo Inmetro



Ângulo, em graus, que a escada forma com o chão ao ser apoiada na parede

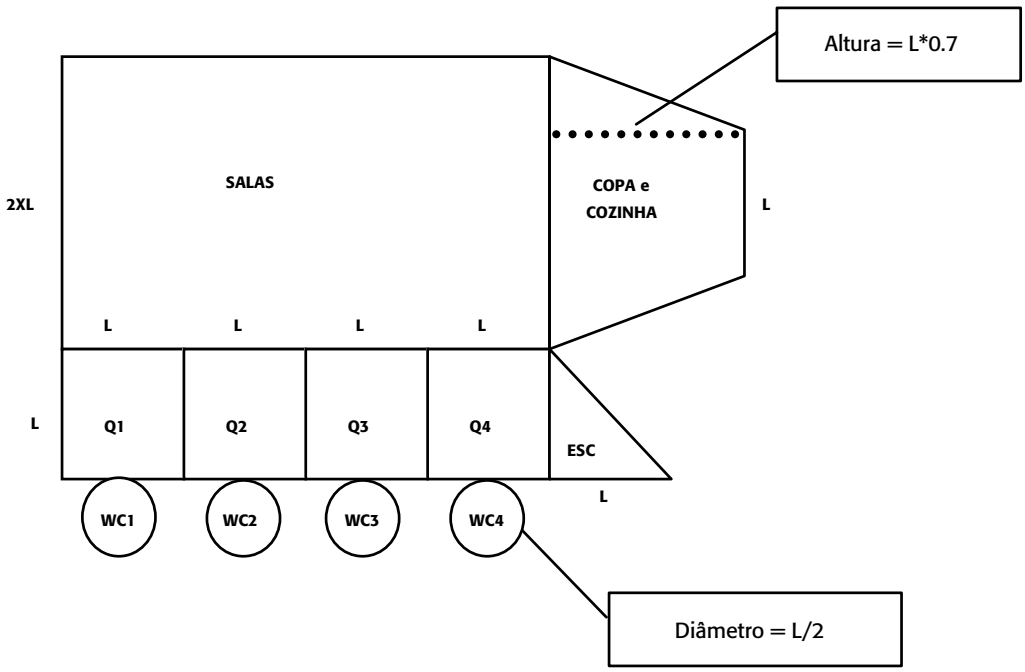
13.4 Desafio 4

Um problema típico em ciência da computação consiste em converter um número da sua forma decimal para a forma binária. Por exemplo, o número 12 tem sua representação binária igual a 1.100. A forma mais simples de fazer isso é dividir o número sucessivamente por 2, cujo resto da *i*-ésima divisão vai ser o dígito *i* do número binário (da direita para a esquerda).

Por exemplo: 12 / 2 = 6, resto **0** (1º dígito da direita para esquerda), 6 / 2 = 3, resto **0** (2º dígito da direita para esquerda), 3 / 2 = 1 resto **1** (3º dígito da direita para esquerda), 1 / 2 = 0 resto **1** (4º dígito da direita para esquerda). Resultado: **12 = 1100.**

13.5 Desafio 5

Um escritório de engenharia está fazendo os cálculos de quanto gastará com a mão de obra dos seus pedreiros. Sabe-se que cada pedreiro ganha 10% do valor do salário mínimo por metro quadrado de construção. Logo, para a construção de um condomínio fechado com 40 casas, tem-se, para cada casa, apenas um pedreiro trabalhando, e cada casa tem a planta a seguir. Calcular e mostrar o custo da mão de obra do condomínio.



13.6 Desafio 6

Faça um programa que receba um número entre 1 e 10.000 e mostre o valor digitado por extenso.

digite um número: 10.500
número inválido
digite um número: 130
cento e trinta

digite um número: 5
cinco

digite um número: 1.259
um mil e duzentos e cinquenta e nove

13.7 Desafio 7

Faça um programa que simule um jogo de forca.

Restrições:

1. Devem ser previamente cadastradas dez palavras.
2. Com cada palavra deve ser cadastrada uma dica.
3. A palavra a ser acertada deve ser sorteada.
4. Quando o usuário tiver só mais uma chance, a dica deve ser apresentada.

5. No total, o usuário deve ter no máximo sete chances.
6. Letras maiúsculas e minúsculas devem ser aceitas.
7. O usuário pode jogar quantas vezes quiser.
8. As letras já testadas devem ser mostradas e não podem ser testadas mais de uma vez.

13.8 Desafio 8

Faça um programa que apresente o menu de opções a seguir:

1. **C**adastrar pessoa na agenda de aniversariantes (nome, dia e mês do aniversário).
2. Excluir pessoa a partir do nome.
3. Alterar dia ou mês a partir do nome.
4. Consultar aniversariantes de uma data (dia e mês).
5. Consultar aniversariantes por mês.
6. Consultar aniversariantes pela letra inicial do nome.
7. Mostrar toda a agenda ordenada pelo nome.
8. Mostrar toda a agenda ordenada por mês.
9. Sair.

A agenda pode suportar até 15 pessoas.

13.9 Desafio 9

Faça um programa que carregue três vetores nos quais serão armazenados os códigos, os salários e o tempo do serviço, em anos, de cinco funcionários. Posteriormente, o programa deverá:

a) Receber um valor que corresponde ao salário a ser consultado e, em seguida, mostrar **dois relatórios**. Cada relatório deverá mostrar o código do funcionário e o salário. O primeiro relatório deve mostrar os funcionários que têm salário até o valor digitado e o segundo relatório deve mostrar os funcionários que possuem salário superior ao valor digitado. Caso não exista nenhum funcionário em algum dos relatórios, mostrar mensagem.

b) Encontrar o menor salário pago, calcular e mostrar **quantos** funcionários têm salário igual ao menor salário e, posteriormente, **mostrar** os códigos desses funcionários.

c) **Gerar** e **mostrar** um quarto vetor com os códigos dos funcionários que possuem tempo de serviço entre 2 e 4 anos e são isentos de impostos. Sabe-se que os funcionários isentos de impostos são aqueles que possuem salário inferior a R$ 1.500,00. Caso nenhum funcionário preencha os requisitos, mostrar mensagem.

13.10 Desafio 10

Faça um programa que apresente o menu de opções a seguir:

1. Incluir um número no vetor.
2. Consultar todos os números do vetor.
3. Consultar um número do vetor.
4. Excluir um número no vetor.
5. Esvaziar o vetor.
6. Sair.

Todas as operações anteriores devem ser realizadas em um vetor de dez posições e o vetor pode conter números repetidos, que podem ser desordenados.

13.11 Desafio 11

Faça um programa que apresente o menu de opções a seguir:

1. Incluir um número no vetor.
2. Consultar todos os números do vetor.
3. Consultar um número do vetor.
4. Excluir um número no vetor.
5. Esvaziar o vetor.
6. Sair.

Todas as operações anteriores devem ser realizadas em um vetor de dez posições e o vetor não pode conter números repetidos, que devem estar ordenados.

13.12 Desafio 12

Uma empresa do ramo da construção civil está informatizando seu Departamento de Pessoal.
Inicialmente, cadastrou o salário de todos os cargos da empresa.

	1	2	3	4	5
Cargos	R$ 2.500,00	R$ 1.500,00	R$ 10.000,00	R$ 1.200,00	R$ 800,00

Cada tipo de cargo ocupa uma posição do vetor de tamanho 5.
Depois, cadastrou todos os seus funcionários em um vetor de registros, contendo os seguintes campos: código, nome e código do cargo.

	1	2	3	4	5	
codigo	15	1	26	12	8	...
nome	João da Silva	Pedro Santos	Maria Oliveira	Rita Alcântara	Lígia Matos	...
codigo_cargo	1	2	3	5	2	...

Crie uma aplicação que contenha uma função para mostrar um menu ao usuário, assim:

1. Cadastrar os cargos da empresa.
2. Cadastrar os funcionários da empresa.
3. Mostrar um relatório contendo o número, o nome e o valor do salário de todos os funcionários.
4. Mostrar o valor pago aos funcionários que pertençam a um cargo informado pelo usuário.
5. Finalizar.

Opção 1: Cada vez que essa opção for selecionada deverá ser chamada uma sub-rotina, na qual o usuário poderá cadastrar todos os cargos. Não se esqueça, nessa empresa existem apenas cinco cargos. Se o usuário mandar executar esta opção mais de uma vez, mostre a mensagem de erro "Salários dos cargos já cadastrados" e retorne ao menu.

Opção 2: Cada vez que essa opção for selecionada deverá ser chamada uma sub-rotina, na qual o usuário poderá cadastrar um novo funcionário, ou seja, informará o número do funcionário (este número deve ser único, você deverá implementar essa validação), nome e código do cargo (lembre-se de que o código informado deverá existir no vetor de cargos). Não se esqueça, nessa empresa existem apenas 15 funcionários. Se o usuário selecionar essa opção e o vetor de funcionários estiver completamente preenchido, mostrar uma mensagem de erro e retornar ao menu.

- Crie uma sub-rotina para fazer a validação do número do funcionário — ela não poderá aceitar número repetido.

- Crie uma sub-rotina para validar o nome do funcionário, obrigando-o a ser composto por pelo menos duas palavras (nome e sobrenome).
- Crie uma sub-rotina para fazer a validação do código do cargo ocupado pelo funcionário — ela só poderá aceitar códigos entre 1 e 5 cujos salários já tenham sido cadastrados no vetor de cargos.

Opção 3: Cada vez que essa opção for selecionada deverá ser chamada uma sub-rotina, na qual serão mostrados código, nome e valor do salário de todos os funcionários cadastrados (salários podem ser obtidos no vetor de cargos).
Opção 4: Cada vez que essa opção for selecionada deverá ser chamada uma sub-rotina, na qual será feito o somatório do salário de todos os funcionários que pertencerem a determinado cargo. Esse cargo é informado pelo usuário (entre 1 e 5) no módulo principal do seu programa e o somatório calculado deverá ser mostrado, também, no módulo principal.

13.13 Desafio 13

Alguns números inteiros possuem a capacidade de se autoelogiarem através de seus dígitos. Estes são números que formam a família dos Números Narcisistas. Os Números Narcisistas clássicos são aqueles iguais à soma de cada um de seus dígitos elevados à potência do número total de dígitos.

Por exemplo, o número 153 é um narcisista clássico porque a soma de cada um de seus dígitos elevados ao cubo (total de dígitos que compõem o número 153) é exatamente 153.

$$153 = 1^3 + 5^3 + 3^3 = 1 + 125 + 27 = 153$$

Crie um programa que receba um número qualquer e determine se ele é Narcisista ou não.

13.14 Desafio 14

Dois números são considerados AMIGÁVEIS se um deles corresponder à soma dos divisores (exceto o próprio número) do outro.

Por exemplo: vamos analisar os números 8 e 10.

Os divisores de 8 são: 1, 2 e 4, resultando em soma igual a 7.

Já os divisores de 10 são: 1, 2 e 5, resultando em soma igual a 8.

Assim, como a soma dos divisores de 10 (exceto ele próprio) resulta em 8, pode-se dizer que os números 10 e 8 são amigáveis.

Criar um programa que receba dois números inteiros quaisquer e determine se são amigáveis ou não.

13.15 Desafio 15

Uma escola oferece 3.058 cursos. Sabe-se que cada curso possui descrição, quantidade de alunos matriculados e valor da mensalidade. A escola precisa cadastrar os cursos e, depois, precisa saber:

a) a média aritmética de alunos matriculados nos cursos;
b) a descrição do curso que gera a maior receita (receita = quantidade de alunos * valor da mensalidade).

Crie uma aplicação que, utilizando um vetor de registro, consiga atender as necessidades da escola.

13.16 Desafio 16

Uma universidade deseja fazer a apuração do resultado do vestibular dos cursos de Ciência da Computação, Engenharia de Computação e Análise de Sistemas.

Para isso, contabilizará o total de pontos obtidos pelos candidatos e armazenará em uma matriz 3×40, onde a linha representa o curso (1ª linha Ciência da Computação, 2ª linha Engenharia de Computação e 3ª linha Análise de Sistemas).

Cada célula da matriz deverá conter o código do candidato e sua pontuação.

Considerando que cada curso possui apenas quarenta vagas, nessa matriz deverão ficar armazenadas as informações apenas dos quarenta melhores candidatos.

Enquanto o curso possuir menos de quarenta candidatos cadastrados, qualquer inserção será aceita (aconselha-se a inserção em ordem decrescente de pontuação). Quando já existirem quarenta candidatos em determinado curso e for digitada uma pontuação maior que a do último colocado, este deverá ser eliminado para que o novo candidato seja inserido na matriz.

Criar uma aplicação que digite diversas pontuações e, ao final, mostre o código e a pontuação dos quarenta aprovados em cada curso.

13.17 Desafio 17

Uma empresa de telefonia adotou um sistema de cobrança cujo valor do minuto de uma ligação varia de acordo com o horário de início:

- das 0h às 9h — 50% de desconto no valor do minuto.
- das 9h01 às 18h — 0% de desconto no valor do minuto.
- das 18h01 às 21h — 30% de desconto no valor do minuto.
- das 21h01 às 23h59 — 40% de desconto no valor do minuto.

Faça um programa que receba o horário inicial (hora e minuto) e o horário final (hora e minuto), junto com o valor normalmente cobrado pelo minuto de uma ligação.

De acordo com os dados anteriores, aplique corretamente os descontos e mostre o valor normal a ser cobrado e o valor com o desconto concedido.

Observação

Se a ligação começou em uma faixa de desconto e terminou em outra, você deverá calcular o desconto de acordo com o período de duração da ligação em cada faixa. Por exemplo, se uma ligação começou às 8h30 e terminou às 20h, você deverá conceder 50% de desconto para o período de 8h30 até as 9h, 0% de desconto para o período de 9h01 até as 18h e 30% de desconto para o período de 18h01 até as 20h.

13.18 Desafio 18

Você foi contratado para trabalhar em uma empresa de segurança e está encarregado de registrar os nomes de várias pessoas suspeitas de corrupção.

Temendo que esse arquivo caia em mãos erradas, você decidiu que as informações deveriam ser criptografadas antes da gravação e somente pessoas autorizadas possuirão mecanismos para descriptografá-las.

Assim, implemente uma aplicação que grave em um arquivo de texto uma quantidade indeterminada de dados, no seguinte padrão:

caractere1; caractere2; informação1;informação2&
caractere1; caractere2; informação1;informação2;informação3&

O símbolo ; separa dados de uma mesma pessoa. O símbolo & separa uma pessoa de outra.

O objetivo dessa aplicação é fazer o trabalho de criptografar/descriptografar.

Para cada pessoa, solicite o caractere1 e o caractere2. Eles serão a base para o processo de criptografar/descriptografar.

Quando as informações de uma pessoa forem recebidas, todos os símbolos contidos na informação, iguais ao caractere2, deverão ser substituídos pelo caractere1 e, só então, a gravação deverá ser realizada no arquivo.

A gravação dos dados termina quando for fornecido o símbolo @ como entrada.

Quando uma leitura for feita, todos os símbolos contidos na informação iguais ao caractere1 deverão ser substituídos pelo caractere2 para, só então, serem mostrados ao usuário.

Mostre todos os dados gravados (descriptografados).

13.19 Desafio 19

Um médico está melhorando o processo de agendamento de consultas, a fim de evitar transtornos decorrentes de esquecimentos de sua secretária.

Basicamente, o médico precisa de:

1. Cadastramento de pacientes (código, nome paciente, nome convênio, telefones fixo e celular).

   Não podem existir pacientes com código repetido. Criar um arquivo, chamado paciente.dat.
2. Agendamento (data, hora, código do paciente, tipo consulta — Normal ou Retorno).

   Não podem existir duas consultas agendadas para o mesmo dia e horário. Criar um arquivo chamado agenda.dat.
3. Alteração de pacientes.

   Informe o código do paciente. Se encontrar paciente gravado no arquivo de Pacientes, altere seus dados. Usar o arquivo paciente.dat gerado no item 1.
4. Visualização de consultas.

   Mostre todas as consultas agendadas (inclusive o nome do paciente). Usar os arquivos paciente.dat e agenda.dat, criados nos itens 1 e 2.
5. Geração de arquivo auxiliar

   Paralelamente ao seu trabalho, outra empresa está desenvolvendo uma aplicação que, no dia da consulta, envia uma mensagem ao telefone celular do paciente, alertando-o sobre o horário agendado com o médico. Você, então, deverá gerar para essa aplicação um arquivo de texto, contendo nome do paciente, data da consulta e horário da consulta (separe cada dado por um hífen).

13.20 Desafio 20

Uma rede de lanchonete está implantando um sistema informatizado de controle de estoque. Cada filial dessa rede segue um padrão rigoroso, em que todos os lanches utilizam os mesmos ingredientes, gerando, assim, a mesma lista de produtos em estoque.

É interessante manter cadastrados descrição, quantidade em estoque, quantidade mínima exigida e valor unitário de cada produto. Cada filial possui seis produtos em estoque.

Para o efetivo cadastramento, os projetistas do sistema sugeriram aos programadores a utilização de um vetor de seis posições para o cadastramento da descrição dos produtos e uma matriz 6 × 3 para cadastrar quantidade em estoque, quantidade mínima exigida e valor unitário.

Matriz de descrição

1	2	3	4	5	6
Pão com gergelim	Alface americana	Tomate	Queijo Cheddar	Hambúrguer	Nuggets

Matriz com valores numéricos

	1	2	3
1	10	5	1.2
2	20	8	2.3
3	15	4	2.7
4	23	15	30.5
5	100	60	3.1
6	200	50	2.8
	Quantidade em estoque	Quantidade mínima exigida	Valor unitário

Seguindo o definido em projeto, você foi contratado para implementar as seguintes funcionalidades, acessadas por meio de um menu de opções:

1. Cadastrar as informações dos seis produtos, atendendo às seguintes restrições:
   a) a quantidade em estoque não poderá ser inferior à quantidade mínima exigida;

b) o valor unitário deverá ser maior que zero — caso seja informado algum valor incorreto, mostre uma mensagem de erro e solicite-o novamente.

2. Retirar um produto do estoque: você deverá informar o nome do produto desejado e a quantidade desejada.

Então, deverá procurar o produto no vetor de descrições. Se ele não existir, mostrar uma mensagem de erro e voltar ao menu de opções. Se o produto existir, você deverá verificar se a quantidade em estoque é suficiente para atender à solicitação. Se a quantidade existente for suficiente, deverá ser atualizada (por exemplo, se o produto pão com gergelim possui 10 unidades em estoque e vou utilizar 3, deverei atualizar a quantidade em estoque para 7). Se a quantidade existente não for suficiente para atender à solicitação, mostrar a mensagem "Estoque insuficiente" e voltar ao menu de opções.

3. Mostrar a descrição de todos os produtos com quantidade em estoque inferior ao estoque mínimo exigido.
4. Mostrar o valor total dos produtos existentes no estoque.

13.21 Desafio 21

Uma empresa necessita criar um software capaz de controlar as vendas realizadas, com o objetivo de gerar alguns relatórios que auxiliem no processo de reposição de estoque.

A empresa detectou que precisa ter acesso rápido a algumas informações:

- quantidade vendida de determinado produto em um período;
- faturamento em determinado período (somatório das vendas realizadas);
- valor recebido em um período;
- produtos com estoque abaixo do mínimo exigido;
- lucro do período (para definir o lucro, deve ser descoberta a quantidade vendida de cada produto para fazer a diferença entre o valor cobrado do cliente e o valor pago ao fornecedor).

Para atender a essas solicitações, uma equipe de analistas definiu a necessidade de criar alguns arquivos para armazenamento permanente de dados, os quais permitirão a geração dos relatórios descritos anteriormente:

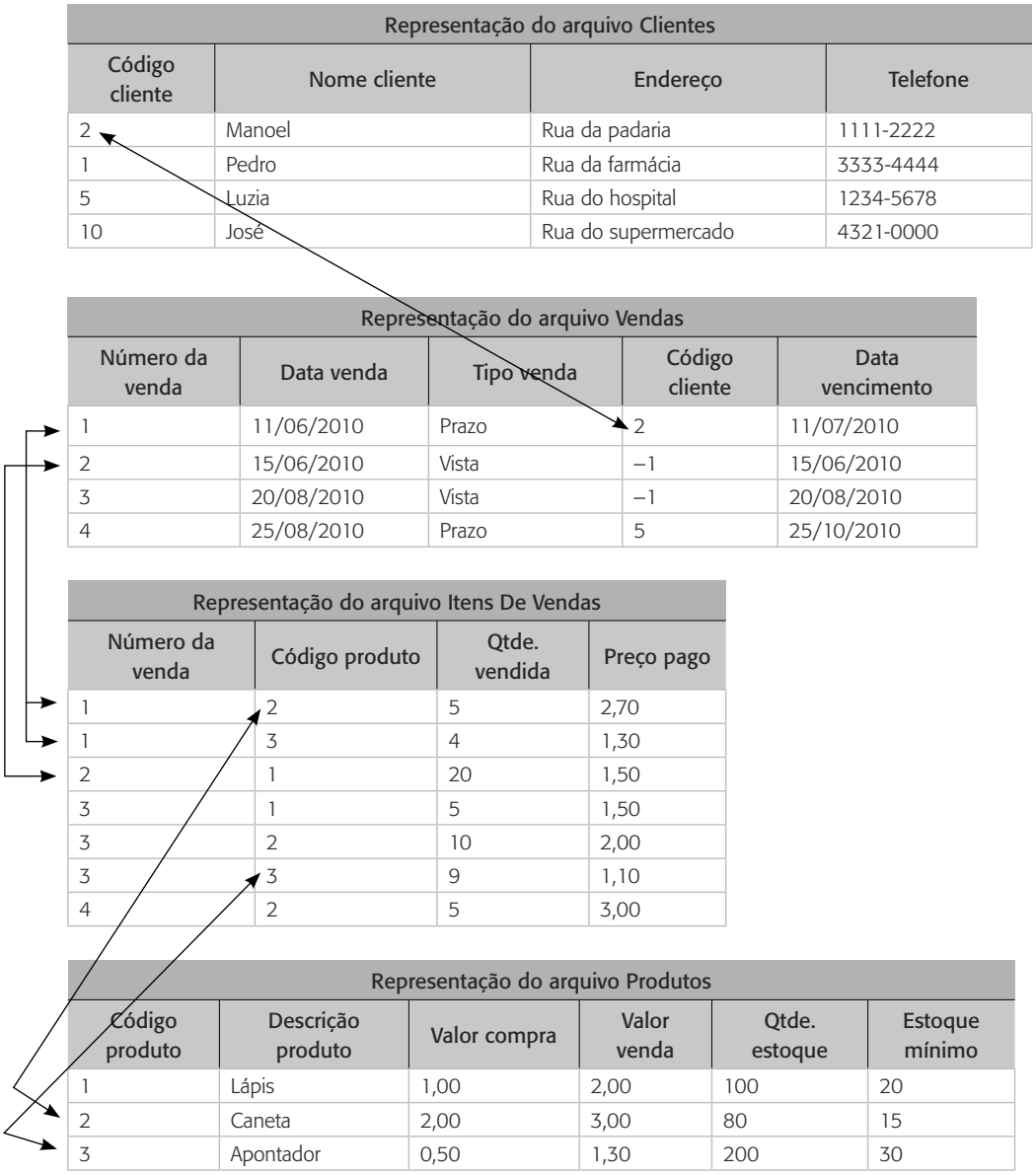
Arquivo Produtos: nesse arquivo, deverão ficar registrados todos os produtos comercializados pela empresa: código único do produto, descrição (String de 30 caracteres), valor de compra, valor de venda, estoque mínimo exigido estoque atual.

Arquivo Cliente: nesse arquivo, deverão ficar registrados os dados dos clientes que podem realizar compras a prazo: código único do cliente, nome do cliente, endereço telefone.

Arquivo Vendas: nesse arquivo, deverão ficar registrados dados das vendas: número da venda, data da venda, tipo da venda (à vista ou a prazo), código do cliente (se a venda for a prazo, preencher esse campo com um cliente válido, caso contrário preencher este campo automaticamente com –1) e data do vencimento (se a venda for à vista, esse campo deve ser preenchido automaticamente com a data da venda, caso contrário, solicitar uma data igual ou superior à data da venda).

Arquivo Item de Vendas: considerando que uma venda pode estar associada a vários produtos, é necessário criar um arquivo que relacione o arquivo Vendas com o arquivo Produtos. Nesse arquivo, deverão ser gravados: número da venda, código do produto, quantidade vendida, valor da venda (para descobrir o valor que um cliente pagou por um produto no passado).

A estrutura dos arquivos pode ser representada conforme mostrado na próxima página:



Representação do arquivo Clientes

Código cliente	Nome cliente	Endereço	Telefone
2	Manoel	Rua da padaria	1111-2222
1	Pedro	Rua da farmácia	3333-4444
5	Luzia	Rua do hospital	1234-5678
10	José	Rua do supermercado	4321-0000

Representação do arquivo Vendas

Número da venda	Data venda	Tipo venda	Código cliente	Data vencimento
1	11/06/2010	Prazo	2	11/07/2010
2	15/06/2010	Vista	–1	15/06/2010
3	20/08/2010	Vista	–1	20/08/2010
4	25/08/2010	Prazo	5	25/10/2010

Representação do arquivo Itens De Vendas

Número da venda	Código produto	Qtde. vendida	Preço pago
1	2	5	2,70
1	3	4	1,30
2	1	20	1,50
3	1	5	1,50
3	2	10	2,00
3	3	9	1,10
4	2	5	3,00

Representação do arquivo Produtos

Código produto	Descrição produto	Valor compra	Valor venda	Qtde. estoque	Estoque mínimo
1	Lápis	1,00	2,00	100	20
2	Caneta	2,00	3,00	80	15
3	Apontador	0,50	1,30	200	30

Seu trabalho como programador é implementar uma aplicação, usando arquivos, que permita mostrar um menu de opções, para que o usuário decida o que deseja fazer: manutenção no arquivo de produto, manutenção no arquivo de cliente, manutenção no arquivo de venda, realizar consultas, ou encerrar a execução da aplicação.

1. **Para manutenção de produtos**

Fornecer as seguintes opções:

- **Cadastrar novo produto:** gerar o código único automaticamente, receber as demais informações do produto e gravar tudo no arquivo correspondente.
- **Consultar produto:** solicitar que o usuário informe o código do produto desejado e buscá-lo no arquivo. Caso o encontre, mostrar todas as suas informações. Caso não o encontre, mostrar uma mensagem de erro e retornar ao menu Manutenção de Produtos.
- **Excluir produto:** solicitar que o usuário informe o código do produto desejado e buscá-lo no arquivo. Caso não o encontre mostre mensagem de erro. Se o encontrar, verifique, então, se tal produto foi usado em alguma venda. Se sim, mostre uma mensagem informando que a exclusão não poderá ser realizada. Se o produto não estiver vinculado a nenhuma nota, efetive a exclusão.

- **Alterar produto:** solicitar que o usuário informe o código do produto desejado e buscá-lo no arquivo. Caso ele não exista, mostrar mensagem de erro. Caso exista, sobrepor todos os dados com os novos valores fornecidos pelos usuários.

2. **Para manutenção de clientes**

Disponibilizar as mesmas funções do menu de Manutenção de produtos.

3. **Para realização de vendas**

Gerar o número da Nota Fiscal de venda automaticamente. A data da emissão da Nota fiscal deverá ser capturada do sistema. O tipo da venda deverá ser fornecido pelo usuário, podendo ser "à vista" ou "a prazo".

Caso o usuário opte por venda a prazo, deverão ser fornecidos o código do cliente (que deverá ter sido previamente cadastrado no arquivo de clientes) e data de vencimento (que deverá ser igual ou superior à data da emissão da Nota Fiscal).

Caso as informações estejam todas corretas, gravar os dados no arquivo Notas e permitir que o usuário cadastre diversos produtos nessa nota (gravando-os no arquivo ItensDeVendas).

O usuário fornece o código do produto (que deverá ter sido previamente cadastrado no arquivo de produtos) e a aplicação mostra a descrição correspondente.

Não permita vender quantidade maior que a registrada no arquivo de produtos. Ao confirmar a venda, atualizar o estoque.

4. Consultas

Permitir que sejam realizadas as seguintes consultas:

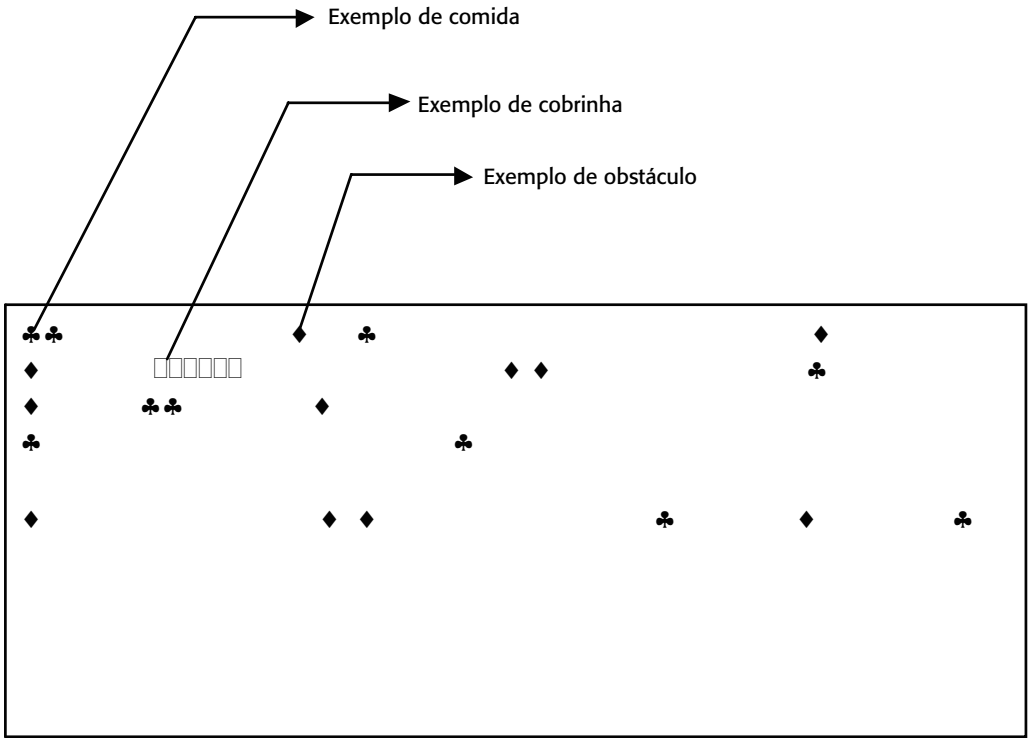
- Número das vendas realizadas em determinado período informado pelo usuário.
- Código e descrição dos produtos com quantidade em estoque abaixo do estoque mínimo permitido.

13.22 Desafio 22

Crie uma aplicação que simule o jogo descrito a seguir:

É o conhecido jogo da cobrinha com algumas pequenas alterações.

Você deverá criar uma janela com as dimensões que desejar. Dentro dessa janela deverão ser colocados dois tipos de elementos: comida para a cobrinha e obstáculos que ela terá de transpor.



O início do jogo

Você precisará distribuir, em posições geradas aleatoriamente, 10 obstáculos e 15 comidas.

A direção de movimento inicial da cobrinha deverá ser gerada aleatoriamente pelo programa (para cima, para baixo, para esquerda ou para direita).

Você deverá estipular um tempo de duração do jogo.

O jogador começará o jogo com cinco vidas.

Movimento da cobrinha

A cobrinha se movimentará em uma direção até que o usuário aperte uma das teclas de direção (←↑→↓) para alterar sua trajetória.

Atenção: A cobrinha não pode passar por cima dela mesma, por isso:

- se a cobrinha estiver subindo, não aceite que o usuário aperte a tecla ↓;
- se a cobrinha estiver descendo, não aceite que o usuário aperte a tecla ↑;
- se a cobrinha estiver indo para a esquerda, não aceite que o usuário aperte a tecla →;
- se a cobrinha estiver indo para a direita, não aceite que o usuário aperte a tecla ←.

A cobrinha não poderá ultrapassar os limites da janela definida por você.

Então, se a cobrinha:

- estiver subindo, chegar à borda, e se sua trajetória não for alterada, deverá ir desaparecendo da parte superior e ir aparecendo na parte inferior da janela (mantendo a mesma coluna);
- estiver descendo, chegar à borda, e se sua trajetória não for alterada, deverá ir desaparecendo da parte inferior e ir aparecendo na parte superior da janela (mantendo a mesma coluna);
- estiver indo para a esquerda, chegar à borda, e se sua trajetória não for alterada, deverá ir desaparecendo da lateral esquerda e ir aparecendo na lateral direita da janela (mantendo a mesma linha);
- estiver indo para a direita, chegar à borda, e se sua trajetória não for alterada, deverá ir desaparecendo da lateral direita e ir aparecendo na lateral esquerda da janela (mantendo a mesma linha).

A comida

Quando a cabeça da cobrinha atingir a mesma coordenada de uma comida, esta será ingerida e deverá sumir da tela. Para cada três comidas ingeridas, o jogador ganha uma vida.

O obstáculo

Quando a cabeça da cobrinha atingir a mesma coordenada de um obstáculo, este deverá sumir da tela. Para cada obstáculo atingido, o jogador perde uma vida.

O fim do jogo

O jogo termina quando:

1. a cobrinha comer todas as comidas. Nesse caso, o jogador ganhou o jogo;
2. acabaram as vidas. Nesse caso, o jogador perde o jogo;
3. o tempo acabou. Nesse caso, o jogador perde o jogo.

Dica: Pesquise funções que detectam qual tecla foi pressionada em determinado momento.

13.23 Desafio 23

Um banco possui vários tipos de contas bancárias: (1) conta-corrente simples; (2) conta-corrente especial; (3) conta poupança.

Todas as contas possuem um número, um titular e um saldo. Para cada uma delas, entretanto, existem peculiaridades.

Conta corrente simples: tem direito a um cartão de débito (guardar o número) e um talão de cheques (guardar o número do primeiro e do último cheque do talão), mas não tem direito a limite e nem cheque especial;

Conta corrente especial: tem tudo o que a conta simples possui, mais o limite de crédito concedido pelo banco e taxa de juros cobrada pelo uso do limite;

Conta poupança: tem tudo o que a conta simples possui, mais uma data de aniversário (o dia do mês em que os rendimentos são creditados).

Defina as classes anteriores, utilizando herança onde for apropriado.

Crie uma aplicação que gere uma conta de cada tipo e cadastre (nos métodos construtores) as informações pertinentes a cada uma das contas.

Depois que as contas estiverem cadastradas, seu programa deverá mostrar um menu com opções para visualização das contas e para atualização do saldo de cada conta, lembrando que: para as contas simples, o saldo está sempre atualizado; para as contas especiais, o valor do limite utilizado deverá ser subtraído do saldo, e, para as contas poupanças, o saldo deverá ser acrescido do rendimento (vamos supor 2% ao mês).

13.24 Desafio 24

Um promoter deseja controlar todos os eventos dos quais participa e você foi contratado para resolver esse problema.

Assim que começou o trabalho, você ficou sabendo que o cadastro de um evento necessita de: identificador único (um código), descrição, local e data de realização, quantidade de convites colocados à venda, custos de organização e valor da entrada.

Foi-lhe informado, também, que existem sempre três valores de entrada: o 1º valor é para as entradas do tipo popular, o 2º valor é para as entradas do tipo normal e o 3º valor é para as entradas do tipo VIP.

Alguns desses eventos são festas open bar, possuindo, assim, além de todas as informações anteriores, uma relação das quatro bebidas que serão servidas. As informações da bebida, por sua vez, são nome, teor alcoólico e valor unitário.

O valor das entradas populares pode ser informado pelo usuário ou pode ser calculado automaticamente, da seguinte forma:

- eventos open bar: R$ 30,00 mais 50% do valor unitário de cada bebida que será servida;
- demais eventos: custo de organização dividido pela quantidade de convites colocados a venda.

O valor da entrada normal é o valor da popular + 10% desse mesmo valor. O valor da entrada VIP é o valor da normal + 15% desse mesmo valor.

Implemente classes que representem esse contexto. Utilize, onde apropriado, todos os conceitos de orientação a objetos já estudados (herança, composição, polimorfismo, sobrecarga e sobreposição de métodos etc.).

Crie, também, uma classe aplicação que permita o cadastramento de cinco eventos (que podem ser open bar ou não, dependendo do usuário).

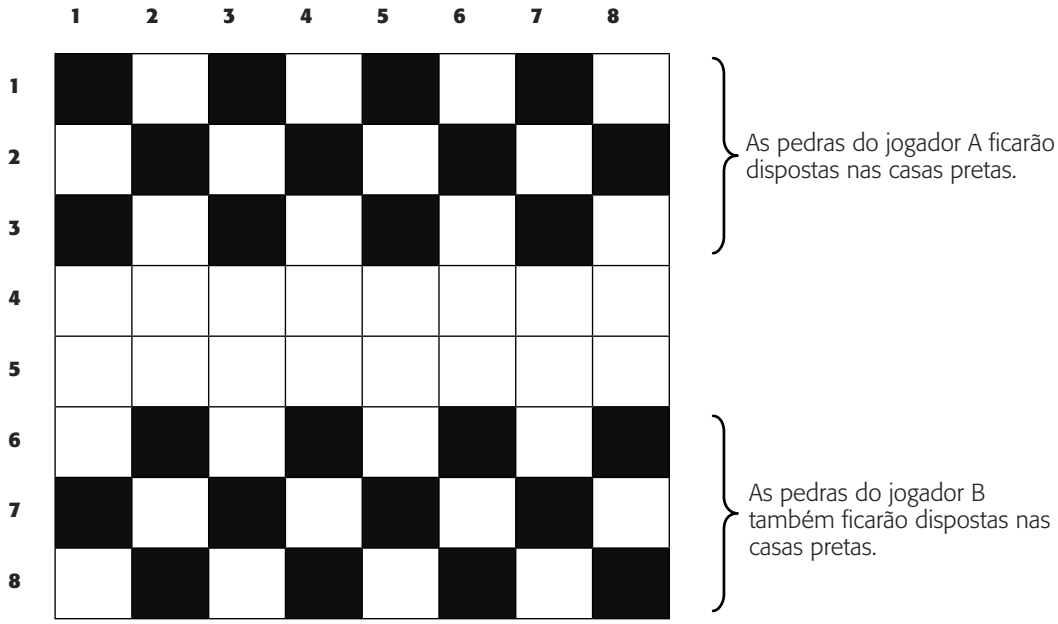
Durante o cadastramento, deverão ser calculados, nas classes apropriadas, os valores das entradas, de acordo com o desejo do usuário e seguindo as regras de negócio, descritas anteriormente.

Mostre, quando o usuário desejar, todas as informações de todos os eventos cadastrados.

13.25 Desafio 25

Implemente um jogo que siga as regras descritas a seguir:

1. O jogo deverá ser jogado em um tabuleiro com 64 casas (8×8).
2. As pedras sempre são dispostas nas casas de cor escura. Como, nesse jogo, não haverá recursos visuais, as pedras dos participantes deverão ser dispostas de tal forma que consigam percorrer as mesmas diagonais. Sugere-se uma disposição conforme a figura a seguir.



3. As pedras dos jogadores deverão ser representadas na tela por caracteres diferentes (por exemplo, para o jogador A utilizar o símbolo O e para o jogador B utilizar o símbolo X).
4. As pedras devem se movimentar sempre em diagonal e para a frente (exceto quando for tomar — comer — uma pedra do adversário, situação em que é permitido movimentar-se para trás).
5. Quando uma pedra atravessar todo o tabuleiro, será coroada como "dama". Essa pedra deverá ser destacada das demais com um tipo diferente de caractere.
6. Em cada jogada, o jogador tem obrigação de tomar — "comer" — todas as pedras possíveis do seu adversário. Caso isso não seja feito ocorrerá um "sopro", o jogador perderá a pedra que não realizou todas as tomadas necessárias.
7. Uma pedra simples pode mover-se apenas uma casa por vez. Exceto quando for comer outra pedra, quando acabará movimentando-se duas casas.
8. Uma pedra comum poderá comer uma pedra do adversário se, e somente se, conseguir pular esta pedra (por exemplo: a pedra localizada na linha 3 coluna 1 poderá comer a pedra da linha 4 coluna 2).
9. Uma dama poderá percorrer várias casas em uma diagonal. Assim, poderá comer pedras que estejam distantes dela (por exemplo, uma dama na linha 4 coluna 2 poderá comer uma pedra na linha 7 coluna 5, reposicionando-se na linha 8, coluna 6).
10. O salto de uma dama só é impedido por uma obstrução, ou seja, quando na mesma diagonal houver outra pedra do mesmo jogador ou duas ou mais pedras do adversário em posições contíguas.
11. As tomadas podem ser simples ou em cadeia. Tomada simples é aquela na qual apenas uma pedra é comida. Tomada em cadeia é aquela em que várias pedras são comidas numa mesma jogada, ou seja, ao término de uma tomada verifica-se a possibilidade de realizar outra e, assim, sucessivamente.
12. As jogadas acontecem alternadamente entre os dois jogadores.
13. Para cada jogada, o jogador deverá informar qual pedra deseja movimentar (informar número da linha e da coluna) e para onde deseja movimentá-la (mais uma vez, informará o número da linha e o da coluna).
14. Depois disso, seu programa deverá fazer todas as validações necessárias, de acordo com as regras apresentadas anteriormente.

15. Será campeão o jogador que acabar com todas as pedras do adversário ou deixá-las sem condição de jogo.
16. Acontecerá um empate quando houver vinte jogadas envolvendo apenas damas, sem que haja tomadas, ou seja, nenhuma pedra é "comida".

Observação

Essa aplicação deverá seguir os conceitos da orientação a objetos. Isso quer dizer que vocês deverão identificar quais classes estão presentes nesse problema.

Após delimitar as classes, definir quais são as responsabilidades de cada uma.