Prova 2

Algoritmos e Estruturas de Dados I

Professor: Pedro O.S. Vaz de Melo

Nome: _____

escrevendo o meu nome eu juro que seguirei o código de honra

Código de Honra para este exame:

- Não darei ajuda a outros colegas durante os exames, nem lhes pedirei ajuda;
- não copiarei nem deixarei que um colega copie de mim;
- não usarei no exame elementos de consulta não autorizados.

Referências:

Função/Operador	Descrição	Exemplo
`rand()`	gera um número aleatório inteiro entre 0 e `RAND_MAX`	rand() pode gerar 41
`RAND_MAX`	o maior número possível que pode ser gerado por `rand()`	`RAND_MAX` é sempre 32767

1. (*8 points*) Escreva uma função que recebe uma *string* `str` contendo apenas letras minúsculas como parâmetro e imprime, em ordem alfabética, todas as letras do alfabeto que não estão presentes em `str`. Exemplo: se a string for `zbxxyimkjowiejowfsjdfqskdhbuudejkt`, o seu programa deve imprimir `a c g l n p r v`. Dica 1: as letras minúsculas têm código ASCII entre 97 (letra `a`) e 122 (letra `z`). Dica 2: lembre do problema que discute uma forma inteligente de guardar sapatos em um armário a partir da numeração dos mesmos.

Importante: nas questões a seguir, você vai implementar um simulador de compras de uma loja semelhante ao que foi passado em sala de aula. A diferença é que os produtos terão estoque limitado e clientes poderão realizar mais de uma compra. O seu programa deve sortear o número de clientes, número de produtos disponíveis na loja, estoque de cada produto, preço de cada produto, número de compras de cada cliente e produtos que cada cliente comprou. Depois disso, deve imprimir o faturamento da loja naquele dia. Para resolver as questões considere as definições dadas na última questão desta prova.

2. (*2 points*) Defina um **novo tipo de dados** para representar um `Produto` e que deve ser capaz de armazenar as seguintes informações: id (um número inteiro), estoque (número de itens em estoque) e preço (com parte fracionária, ex: $9,99$).

3. (*2 points*) Escreva uma função de nome `randFloat` que retorna um número ponto flutuante aleatório entre um valor mínimo (`min`) e máximo (`max`). A sua função deve ter o seguinte protótipo:

```
float randFloat(int min, int max);
```

4. (*2 points*) Implemente uma função de nome `initProduto` que recebe um `Produto` **por referência** e um identificador `cod` por valor. A função deve inicializar os campos do produto da seguinte forma: `id` deve receber `cod`, `estoque` deve receber um número aleatório entre 0 e `MAXESTOQUE` e `preco` um valor aleatório entre 1.00 e 100.00, de forma que os centavos também sejam preenchidos aleatoriamente.

5. (*4 points*) Implemente uma função de nome `estoqueLoja` que recebe como parâmetros um vetor de `Produtos` e a quantidade de produtos que a loja vende. A função deve retornar o número total de produtos que estão em estoque na loja. Protótipo:

```
int estoqueLoja(Produto p[], int n);
```

6. (*6 points*) Implemente uma função de nome `simulaCompra` que recebe como parâmetros um vetor de produtos e o número de produtos diferentes que a loja vende. A sua função deve simular as compras de um cliente e **retornar** o valor total pago pelo cliente. Primeiro, sorteie o número de produtos que o cliente comprará (0 a `MAXCOMPRAS`). Depois, para cada compra, sorteie o código do produto que o cliente quer comprar. O produto será faturado se ele estiver em estoque. Se não tiver em estoque, sorteie um novo código de produto para o cliente. Você deve garantir que a sua função termine corretamente quando o número total de produtos em estoque na loja seja 0 (dica: use a função `estoqueLoja`).

7. (*6 points*)
Escreva um programa para simular e imprimir o faturamento total da loja ao fim de um dia de funcionamento. O seu programa deve criar um vetor de produtos e inicializar cada um deles a partir da função `initProduto`. O número de produtos distintos vendidos na loja (1 a `MAXPRODUTOS`) e o número de clientes (0 a `MAXCLIENTES`) devem ser gerados aleatoriamente. Ao final, o seu programa deve imprimir o valor total vendido. Use as definições listadas abaixo e preocupe-se em implementar apenas o bloco do `main`.

```
#include <stdio.h>
#include "prova2.h"
#define MAXCLIENTES 50 //numero maximo de clientes
#define MAXPRODUTOS 1000 //numero maximo de produtos diferentes que a loja vende
#define MAXESTOQUE 3 //estoque maximo de cada produto
#define MAXCOMPRAS 20 //numero maximo de produtos que cada cliente pode comprar

void main() {

   //o seu código será copiado aqui:
}
```