Universidade Federal do Espírito Santo
Programação II – 2021/2 - EARTE
Prova 2 – 22/03/2022

ATENÇÃO!

A solução da questão deve ser um programa fonte em C contido em um único arquivo texto de extensão .c, nomeado rigorosamente como o formato a seguir:
« ONIBUS_NumeroMatricula.c »

- **O arquivo texto deve ser submetido no AVA da disciplina, no campo correspondente à questão resolvida.**
- **Colocar na caixa de texto apenas considerações que julgue importante.**
- **O valor percentual de cada solicitação está definido em cada item.**
- **Penalidades:**
  - **Código sem identação correta;**
  - **Não usar estruturas de dados corretamente;**
  - **Acesso direto a membros internos de estruturas;**
  - **Uso de repetições desnecessárias;**
  - **Não tratar entrada do usuário;**
  - **Uso de bibliotecas, além da math.h**

ONIBUS: Considere um sistema de transporte público realizado por ONIBUS drones que podem carregar passageiros pela cidade. Por voarem, os ONIBUS DRONES podem ir de um ponto ao outro sem trânsito e em linha reta.

Uma empresa possui **n** ônibus drones que podem navegar pela cidade e transportar pessoas. O valor de **n** pode ser **no máximo** igual a **30**. As rotas de cada ônibus são escolhidas aleatoriamente na inicialização do programa. O objetivo do programa é gerar um relatório do lucro de cada ônibus, isto é, os ônibus que levaram mais passageiros pela menor distância. Para isto:

Cada ônibus efetua um deslocamento relativo em relação ao seu ponto de partida. Essa área é representada por um quadrado que possui seu vértice superior esquerdo localizado no ponto **(0,0)** e seu vértice inferior direito localizado no ponto **(100,100)**. O ponto de partida e de retorno de todos os ônibus é o **ponto de origem (0,0)**.

Cada ônibus passará por no máximo **4 pontos distintos** na área da cidade. Em cada ponto, se há passageiros, é realizado um pouso e os passageiros podem subir. Sabe-se que cada ponto comporta **no máximo 20** pessoas. Além disso, cada passageiro **paga 100 moedas** para viajar. Ao final de todas as viagens, o programa emite um relatório com as informações de identificação do ônibus, número de passageiros, distância viajada e lucro, ordenado por lucro, em ordem decrescente.

Considerando todas as informações acima, o seu programa deve:

a) (1%) Entrada do usuário: Dois inteiros

a.1) Ler pelo teclado o valor de semente de **s, s <= 1000**, onde s é um inteiro que será usado como semente de geração de números aleatórios (*srand(s)*).

a.2) Ler pelo teclado o valor de **n**, **n <= 30**, onde **n** é o número de ônibus drones que irão viajar pela cidade;

b) (9%) Tipos definido pelo usuário obrigatórios*:

tPosicao, representando as coordenadas *x* (tipo *float*) e *y* (tipo *float*) da região que o ônibus passará e a quantidade de passageiros naquele ponto;

tOnibus guarda informações relevantes de um ônibus, que são: seu identificador id (do tipo *int*), a lista dos 4 pontos que ele irá visitar, representada por um vetor do tipo **tPosicao** de dimensão 4, e um tipo para armazenar estatísticas **tStats**.

tStats registra informações adicionais sobre um ônibus. Esse tipo deve conter no mínimo 2 atributos: o total *totp* de passageiros carregados pelo onibus e a distância total *totd* percorrida.

*Você pode definir outros tipos e/ou adicionar outras informações que julgar necessário.

c) (15%) Inicialização:

A função main deve conter um vetor do tipo *tOnibus*, de dimensão 30;

c.1) criar a função *preencheOnibus* para preencher as *n* primeiras posições do vetor de *tOnibus* com o seguinte protótipo:

void preencheOnibus(tOnibus r[], int n)

i) os *n* ônibus devem ser identificados pelo seu id (um valor entre **1** e **n**);

c.2*)* A função que inicializa o tipo tPosicao deve ter protótipo:

tPosicao inicializatPosicao()

As coordenadas dos pontos, assim como o número de passageiros em cada ponto, devem ser valores gerados aleatoriamente. As coordenadas *x* e *y* devem ser valores reais no intervalo [0,99]. Para definir o número de passageiros em cada ponto deve se utilizar a seguinte regra: Há 50% de chance de ter passageiros no ponto, em caso de ter, o número de passageiros será um valor inteiro gerado aleatoriamente entre 0 e 20.

A função para geração de números aleatórios é o rand(). Por exemplo, para gerar valores reais no intervalo [0,1], basta usar a expressão (rand() / (float) RAND_MAX). Essa função gera uma distribuição normal entre os valores.

d) (25%) Crie as funções *distanciaTotal* e *totalPassageiros* para calcular a distância total percorrida e o número total de passageiros em uma rota, respectivamente. Em seguida, crie uma função que retorne o tipo tStats com as informações preenchidas. As funções devem ter os seguintes protótipos

float distanciaTotal(tPosicao p[], int np)

int totalPassageiro(tPosicao p[], int np)

tStats calculaDados(*tPosicao p[], int np)*

sendo *np* o tamanho do vetor de tPosicao.

Importante: O ônibus parte do ponto (0,0), percorre os pontos na sequencia do vetor de *tPosicao* e sempre retorna para o ponto de origem (0,0).

e) (5%) Crie a função que preenche os dados de rota calculada por cada onibus. Protótipo:

void preencheDados(tOnibus r[], int n)

sendo *r* o vetor de *tOnibus* e *n* o tamanho do vetor utilizado.

f) (30%) Crie uma função que tenha como resultado a lista de ônibus ordenada, em ordem decrescente, pelo lucro de cada ônibus.

Para isto, considere que o combustível é o que mais afeta o lucro e que 1 unidade de medida consome 1 litro de combustível, que custa *5 moedas por litro*. Cada passageiro paga *100 moedas* para viajar. Logo, o lucro é calculado como o total de passageiros multiplicado pelo custo da passagem menos a distância percorrida multiplacada pelo custo do combustível.

Você pode definir o protótipo da função bem como fazer alterações nos structs para facilitar a forma de fazer esta função.

g) (5%) Crie a função para exibir um relatório após todos os ônibus circularem. A função deve ter o protótipo:

void imprime_Relatorio (tOnibus r[], int n),

sendo *r* o vetor de *tOnibus* e *n* o tamanho do vetor. Deve ser impresso o identificador do ônibus, o número de passageiros, a distância percorrida e o lucro. Todos os campos devem ser separados por um caracter de tabulação ('\t') e cada ônibus separado por uma quebra de linha. Os números reais devem ser impressos com duas casas de precisão.

h) (10%) Os membros internos de um struct devem ser acessados apenas pelas funções que recebem o struct como argumento. Sua responsabilidade criar essas funções adicionais para garantir a qualidade do código.

Entrada: Dois inteiros**:** Um valor de semente *s, s <=100* e o número de ônibus n, n <=30.

Saída: Relatório com todas as informações de um ônibus, ordenada por lucro, em ordem decrescente, conforme instruções do **item g**.

Exemplo:

Exemplo de entrada:	1 10
Exemplo de saída:	7 37 324.16 2079.21 1 20 219.32 903.41 8 23 283.50 882.48 6 21 287.81 660.95 5 14 225.28 273.62 9 17 339.06 4.71 4 13 345.44 -427.20 2 12 387.03 -735.14 3 10 356.32 -781.58 10 8 323.09 -815.45
Exemplo de entrada:	0 50
Exemplo de saída:	Entrada invalida!

*Observação: Devido ao gerador de números aleatórios, o seu resultado pode divergir do exemplo. É seu dever verificar que as contas estão sendo feitas corretamente.

BOA PROVA!!