



Nome: _____

OBSERVAÇÕES:

- Os programas devem ser desenvolvidos utilizando a linguagem de programação C
- Não é permitido o uso de funções que não tenham sido abordadas durante as aulas
- Não serão aceitas soluções que usem funções para manipulação de *strings* (string.h)

QUESTÃO 1) (3,5 PONTOS) Uma empresa de entregas deseja controlar o desempenho de seus entregadores. Cada entregador registra a quantidade de entregas realizadas, o valor total das entregas em reais e a distância total percorrida no mês (em quilômetros). O gerente deseja gerar um relatório que indique os entregadores com maior faturamento, além do faturamento total e da média por entregador. A empresa possui no máximo 100 entregadores. O arquivo texto (entregadores.txt) contém o nome dos entregadores (máximo 20 caracteres), a quantidade de entregas realizadas, o valor total das entregas e a distância percorrida no mês. Por exemplo:

carlos	45	1350.75	320
ana	60	2100.50	410
bruno	30	900.00	180
mariana	50	1800.25	350
joao	25	625.00	150

A partir deste arquivo, deve-se criar um programa que gere um relatório em um arquivo texto (relatorio.txt) com o seguinte formato:

Nr.	Entregador	Entregas	Faturamento	Distância (km)
1	ana	60	2100,50	410
2	mariana	50	1800,25	350
3	carlos	45	1350,75	320
4	bruno	30	900,00	180
5	joao	25	625,00	150

Faturamento total: 6776,50
Faturamento médio por entregador: 1355,30

QUESTÃO 2) (3,5 PONTOS) Uma empresa mantém um registro de seus clientes em um arquivo binário (clientes.dat), que contém um número indefinido de clientes. Cada cliente é armazenado como uma estrutura contendo:

```
typedef struct {  
    int codigoCliente;  
    char nome[51];  
    int codigoCidade;  
} Cliente;
```

As informações das cidades estão armazenadas em outro arquivo binário (cidades.dat) no qual cada registro contém:

```
typedef struct {  
    int codigoCidade;  
    char nome[51];  
    char estado[3];  
} Cidade;
```

Desenvolva um programa que, para cada cliente no arquivo binário (clientes.dat), busque as informações correspondentes à cidade no arquivo binário (cidades.dat). O programa deve gerar um arquivo de saída em formato texto (saida.txt), contendo, para cada cliente: código do cliente, nome do cliente, nome da cidade e estado da cidade. Os campos devem ser gravados no arquivo de saída separados por vírgulas.

101, João Silva, São Paulo, SP

QUESTÃO 3) (3,0 PONTOS) Considere um formato compacto para armazenar as informações de um funcionário, utilizando uma variável inteira sem sinal (`unsigned int`). O formato compacto aloca os bits da variável da seguinte forma, começando pelos bits mais significativos e indo em direção aos bits menos significativos. Os bits mais significativos, que não são utilizados, serão zerados.

- Código do funcionário: um valor de 7 bits que identifica de forma única o funcionário (0 a 127).
- Horas trabalhadas: número de horas trabalhadas no mês, representado por 9 bits (0 a 511).
- Número de atrasos: número de vezes que o funcionário chegou atrasado, representado por 5 bits (0 a 31).
- Status de bonificação: 1 bit que indica se o funcionário receberá bonificação, sendo 1 para sim e 0 para não.

Por exemplo, para um funcionário com código 45, horas trabalhadas 160, número de atrasos 3 e status de bonificação ainda não calculado (bit 0), a representação compactada seria:

10 bits (não usados)	7 bits (Código)	9 bits (Horas Trabalhadas)	5 bits (Atraso)	1 bit (Status)
0000000000	0101011	010100000	00011	0

Crie um programa que leia um arquivo (`funcionarios.dat`) contendo informações de um número indefinido de funcionários, onde o último bit, responsável pelo status de bonificação, está zerado, indicando que ainda não foi calculado. O programa deve calcular o status de bonificação de cada funcionário seguindo a regra: o funcionário receberá bonificação (bit = 1) se tiver trabalhado 150 horas ou mais no mês e registrado menos de 5 atrasos; caso contrário, o bit de bonificação permanecerá 0. O programa deverá gerar um novo arquivo binário de saída (`atualizados.dat`) contendo os mesmos registros do arquivo original, mas com o bit de bonificação devidamente atualizado.

Por exemplo, considerando um exemplo fictício com os funcionários (em texto, para visualização):

```
00000000000101011010100000000110
00000000000101010010000100000110
00000000000101000010100000001100
00000000000101100010100010000100
```

O programa deverá gerar um arquivo binário (`atualizados.dat`) com o conteúdo:

```
00000000000101011010100000000111
00000000000101010010000100000110
00000000000101000010100000001100
00000000000101100010100010000101
```