Universidade Anhembi Morumbi
LAUREATE INTERNATIONAL UNIVERSITIES

Nota: _____

| | | |
|---|---|---|
| **Nome:** | | **Nº Registro Acadêmico:** |
| **Curso:** | **Dia da semana:** Quinta-feira | **Semestre:** |
| **Nome do Professor:** Fernando Esquirio Torres | **Disciplina:** Técnicas de Programação | **Data:** 02/04/2019 |

# PROVA N1

## Instruções

- **Data da Prova: 02/04/2020 as 9h**
- **Você terá um pouco mais de 24 horas para fazer e entregar a prova pelo Blackboard.**
  - **Prazo de entrega: dia 03/04/2020 até as 10:59h**
- **Entregar o arquivo digital editável, extensões válidas: .pdf, .doc, .docx ou .odt.**
- Fonte Times New Roman, Arial ou Calibri. Tamanho: 10 ou 12.
- **Não será permitido a entrega por e-mail, compartilhamento na nuvem e outros meios, pois é uma avaliação oficial, então é preciso usar um meio oficial para registro das atividades entregues.**
- Prova será individual, cada aluno entrega a sua prova.
- Cópia de provas ou questões será considerada plágio e nesse caso terão suas avaliações anuladas. Seja entre alunos da mesma turma ou de turmas diferentes.
- Dentro do arquivo entregue coloque o máximo de informações sobre você que sejam relevantes para identificá-lo como aluno da disciplina: RA, Nome Completo, Turma, dia da semana, etc. **Vide o cabeçalho no início dessa prova**.
- Caso de falta e perda da prova: Apresentar a justificativa (atestado, declaração, etc.) com o motivo da falta (saúde, trabalho, etc.).
  - Para esses casos será marcada uma nova data e o aluno fará uma nova avaliação.
  - Outros motivos serão verificados e avaliados.
- **Instruções para a prova:**
  - Valor total: 10 (dez) pontos.
  - 20 questões, sendo que o aluno deverá escolher no **mínimo** 10 questões para serem respondidas (valor: 1 ponto por questão).
  - Caso o aluno responda mais do que 10 questões, serão consideradas para calcular a nota final as 10 questões que o aluno obteve maior pontuação.
  - Caso o aluno escolha menos do que 10 questões, perderá 1 ponto por questão não escolhida.

## Critérios de Avaliação

As questões dissertativas serão classificadas, avaliadas de acordo com a escala abaixo e a nota atribuída ao percentual obtido:

| | |
|---|---|
| 100% | Questão respondida de forma coerente e elaborada pelo aluno, que abordou todos os assuntos relacionados à questão. |
| 75% | Questão respondida de forma coerente, que abordou parcialmente os assuntos relacionados à questão. Ou copiada ou não de uma fonte de consulta. |
| 50% | Questão respondida de forma não coerente, que abordou pouco dos assuntos relacionados à questão. Ou questões respondidas de forma longa contendo assuntos diversos que são e não são relativos à questão. |
| 25% | Questão respondida de forma não coerente, que não abordou os assuntos relacionados à questão. |
| 0% | Questão não respondida. |

| CRITÉRIOS PARA AVALIAÇÃO DA COMPETÊNCIA DE COMUNICAÇÃO ESCRITA<br>A nota atribuída aos seus textos poderá sofrer descontos relativos à redação: | |
|---|---|
| **-0,1 pela falta do** uso da norma culta da língua portuguesa: ortografia e gramática. | A redação deve utilizar a norma culta, com concordância nominal e verbal, ortografia e pontuação corretas. |
| **-0,1 pela falta de** coerência textual. | O texto deve encadear as ideias de modo coerente, claro, preciso e compreensível. |
| **-0,1 pela falta de** coesão entre as ideias discutidas no texto. | Ideias, frases e parágrafos devem ser ligados fazendo uso de conectores e referências temporais e lógicas. |
| **-0,1 pela falta de** texto legível e boa apresentação**.** | O texto deve ser legível, limpo e organizado. |

**Escolha no mínimo 10 questões das 20 para serem respondidas.**

**Questão 01 -** Escreva um programa que simule o lançamento de dois dados. O programa deverá usar rand para lançar o primeiro dado, e deverá usar rand novamente para lançar o segundo dado. Em seguida, a soma dos dois valores deverá ser calculada. [Nota: como cada dado pode mostrar um valor inteiro de 1 a 6, então, a soma dos dois valores variará de 2 a 12, com 7 sendo o resultado mais frequente; e 2 e 12, os resultados menos frequentes.] A Figura da questão mostra as 36 combinações possíveis dos dois dados.

| | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|
| 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| 2 | 3 | 4 | 5 | 6 | 7 | 8 |
| 3 | 4 | 5 | 6 | 7 | 8 | 9 |
| 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| 5 | 6 | 7 | 8 | 9 | 10 | 11 |
| 6 | 7 | 8 | 9 | 10 | 11 | 12 |

Resultados do lançamento de dois dados.

Seu programa deverá lançar os dois dados 100 vezes. Use um array (vetor) para contar o número de vezes em que cada resultado possível aparece. Imprima os resultados em um formato tabular. Além disso, determine se os resultados são razoáveis; ou seja, existem seis maneiras de somar um 7, de modo que um sexto de todas as jogadas, aproximadamente, deverá ser 7.

**Questão 02 -** Use um array (vetor) para resolver o problema a seguir. Uma empresa paga o salário de seus vendedores com base em uma comissão. O vendedor recebe R$ 200,00 por semana, e mais 9 por cento de suas vendas brutas nessa semana. Por exemplo, um vendedor, que totalize R$ 3.000,00 em vendas em uma semana, receberá R$ 200,00 e mais 9 por cento de R$ 3.000,00, ou seja, R$ 470,00. Escreva um programa em C (usando um array de contadores) que determine quantos vendedores receberam salários dentro de cada um dos seguintes intervalos (suponha que o salário de cada vendedor seja arredondado em um valor inteiro):

a) R$ 200–299
b) R$ 300–399
c) R$ 400–499
d) R$ 500–599
e) R$ 600–699
f) R$ 700–799
g) R$ 800–899
h) R$ 900–999
i) R$ 1000 ou mais

**Questão 03 -** Escreva um programa de jokenpô (pedra papel e tesoura) usando a melhor de três jogadas. O programa deve ler dois caracteres, um de cada jogador, e verificar quem foi o ganhador da rodada, considerando até três rodadas (melhor de três). Caso um jogador vença duas rodadas seguidas o programa para e exibe o vencedor, caso contrário, é realizada próxima jogada, sendo o máximo três rodadas. Os caracteres são "p" para pedra, "f" para papel e "t" para tesoura, onde as regras são:

- pedra vence tesoura
- tesoura vence papel
- papel vence pedra

**Questão 04 -** Strings são tipos de dados utilizados para trabalharmos com textos, entretanto, diferente de outras linguagens, as strings em C exigem um cuidado especial, pois trabalhamos caractere a caractere, posição por posição do vetor ou através das funções disponíveis na biblioteca string.h. Sendo assim, implemente um programa que receba um texto com até 100 caracteres e retorne a quantidade que cada letra do alfabeto aparece nesse texto. Dica: não há necessidade de diferenciar maiúsculas de minúsculas. Utilize um vetor para contabilizar as ocorrências das letras do alfabeto.

**Questão 05 -** Impressão de letras para vários códigos ASCII. Escreva um programa que receba um código decimal ASCII e imprima o caractere correspondente. Caso o caractere não seja imprimível, você deverá retornar a mensagem "O caractere digitado não é imprimível". Esse programa deverá ser repetido até que o valor -1 seja digitado.

**Questão 06 -** Encontre o erro em cada um dos segmentos de programa a seguir. Se o erro puder ser corrigido, explique como fazê-lo.

a)
```
int *number;
printf( "%d\n", *number );
```

b)
```
float *realPtr;
long *integerPtr;
integerPtr = realPtr;
```

c)
```
int * x, y;
x = y;
```

d)
```
char s[] = "este é um array de caracteres";
int count;
for ( ; *s != '\0'; s++)
   printf( "%c ", *s );
```

e)
```
short *numPtr, result;
void *genericPtr = numPtr;
result = *genericPtr + 7;
```

f)
```
float x = 19.34;
float xPtr = &x;
printf( "%f\n", xPtr );
```

g)
```
char *s;
printf( "%s\n", s );
```

**Questão 07 -** Escreva um programa que sorteie um número de 0 a 20 e que permita que o usuário (sem conhecer o número sorteado) tente acertar. Caso não acerte, o programa deve imprimir uma mensagem informando se o número sorteado é maior ou menor que a tentativa feita. Ao acertar o número, o programa deve imprimir a quantidade de tentativas realizadas.

**Questão 08 -** Simulação: a tartaruga e a lebre. Nesse problema, você recriará um dos momentos realmente importantes da história, a saber, a corrida clássica entre a tartaruga e a lebre. Você usará a geração de números aleatórios para desenvolver uma simulação desse evento memorável. Nossos competidores começam a corrida no 'quadrado 1' de 70 quadrados. Cada quadrado representa uma posição possível ao longo do curso a ser percorrido. A linha de chegada fica no quadrado 70. O primeiro concorrente a alcançar ou passar do quadrado 70 é recompensado com um balde de cenouras e alface. O trajeto é percorrido por uma montanha escorregadia, de modo que, ocasionalmente, os concorrentes perdem o contato com o chão.
Há um relógio que apresenta o tempo em os segundos. A cada segundo indicado pelo relógio, seu programa deverá ajustar a posição dos animais de acordo com as regras da Figura da questão.

| Animal | Tipo de movimento | Porcentagem do tempo | Movimento real |
|---|---|---|---|
| Tartaruga | Caminha rapidamente | 50% | 3 quadrados à direita |
| | Escorrega | 20% | 6 quadrados à esquerda |
| | Caminha lentamente | 30% | 1 quadrado à direita |
| Lebre | Dorme | 20% | Não faz nenhum movimento |
| | Dá um salto grande | 20% | 9 quadrados à direita |
| | Escorrega bastante | 10% | 12 quadrados à esquerda |
| | Dá um salto pequeno | 30% | 1 quadrado à direita |
| | Escorrega pouco | 20% | 2 quadrados à esquerda |

igura 7.31 ■ Regras referentes aos movimentos feitos pela tartaruga e pela lebre usados no ajuste de posições.

Use variáveis para acompanhar as posições dos animais (ou seja, os números de posição vão de 1 a 70). O jogo deve começar com os dois animais na posição 1 (ou seja, 'na linha de partida'). Se um animal escorregar antes do quadrado 1, mova-o de volta para o quadrado 1. Gere as porcentagens na tabela anterior produzindo um inteiro aleatório, i, na faixa de $1 \leq i \leq 10$. No caso da tartaruga, execute um 'caminha rapidamente' quando $1 \leq i \leq 5$, um 'escorrega' quando $6 \leq i \leq 7$, ou um 'caminha lentamente' quando $8 \leq i \leq 10$. Use uma técnica semelhante para mover a lebre. Comece a corrida imprimindo

BANG !!!!!
E LÁ VÃO ELES !!!!!

Depois, para cada segundo marcado pelo relógio (ou seja, cada repetição de um loop), imprima uma linha de 70 posições mostrando a letra T na posição da tartaruga e a letra L na posição da lebre. Ocasionalmente, os competidores acabarão no mesmo quadrado. Nesse caso, a tartaruga morderá a lebre e seu programa deverá imprimir AI!!! a partir dessa posição. Todas as posições de impressão diferentes de T, de L, ou do AI!!! (no caso de um empate) devem estar em branco.
Depois que cada linha for impressa, verifique se um dos animais alcançou ou passou o quadrado 70. Nesse caso, imprima o vencedor e termine a simulação. Se a tartaruga vencer, imprima TARTARUGA VENCE!!! É ISSO AÍ!!! Se a lebre vencer, imprima Lebre vence. Marmelada. Se os dois animais vencerem na mesma batida do relógio, você pode favorecer a tartaruga (a 'pobre coitada') ou pode querer imprimir É um empate. Se nenhum animal vencer, realize o loop novamente para simular a próxima batida do relógio.

**Questão 09 -** Escreva um programa que calcule e imprima a soma dos inteiros pares de 51 a 101. Além disso, complete o seu programa para calcular e imprimir o produto dos inteiros ímpares de 0 a 50.

**Questão 10 -** Calculando o valor de p. Escreva um programa que calcule o valor de PI ($\pi$) a partir da série infinita:

$$\pi = 4 - \frac{4}{3} + \frac{4}{5} - \frac{4}{7} + \frac{4}{9} - \frac{4}{11} + \ldots$$

Imprima uma tabela que mostre o valor de PI ($\pi$) aproximado por um termo dessa série, e depois por dois termos, três termos, e assim por diante. Quantos termos dessa série você precisa usar antes de obter 3,14? 3,141? 3,1415? 3,14159?

**Questão 11 -** Eliminação de duplicatas. Use um array (vetor) de subscrito único para resolver o problema a seguir. Leia 20 números, cada um entre 10 e 100, inclusive. À medida que cada número for lido, imprima-o apenas se ele não for uma duplicata de um número já lido. Considere a 'pior das hipóteses': os 20 números são diferentes. Use o menor array possível para resolver esse problema.

**Questão 12 –** Crie uma estrutura de nome Dieta, que servirá para armazenar o nome da comida, o peso de uma porção e o número de calorias. Crie um vetor de estruturas e escreva um programa que solicite os dados de 10 tipos de comida. Após preencher os dados você deverá imprimir de forma formatada (por exemplo: em uma tabela) os valores inseridos.

**Questão 13 -** Um palíndromo é uma cadeia de caracteres que representa a mesma palavra nos sentidos direto e inverso. Por exemplo, "asa" é um palíndromo, porque o inverso dela também é "asa". Faça um programa que leia uma string e diga se esta é ou não um palíndromo.

**Questão 14 -** Escreva um programa em C que faça o seguinte: Leia 30 (trinta) de leituras de temperatura. Uma leitura consiste em dois números: um inteiro entre -90 e 90, representando a latitude na qual a leitura foi efetuada, e a temperatura observada nessa latitude. Imprima uma tabela consistindo em cada latitude e na temperatura média da latitude. Se não existirem leituras em determinada latitude, imprima "sem dados" em vez de uma média. Em seguida, imprima a temperatura média nos hemisférios Norte e Sul (o Norte consiste em latitudes de 1 a 90 e o Sul em latitudes de -1 a -90). (Essa temperatura média deve ser calculada como a média das médias, não como a média das leituras iniciais.) Determine também o hemisfério mais quente. Ao fazer a determinação, use as temperaturas médias em todas as latitudes de cada hemisfério para os quais existem dados tanto para essa latitude como para a latitude correspondente no outro hemisfério. (Por exemplo, se existirem dados para latitude 57, mas não para latitude -57, então a temperatura média para a latitude 57 deve ser ignorada, ao determinar o hemisfério mais quente.)

**Questão 15 -** Escreva uma estrutura para descrever um mês do ano. A estrutura deve se capaz de armazenar o nome do mês, a abreviação em três letras, o número de dias e o número do mês. Implemente um programa que contenha um vetor com 12 estruturas mencionadas anteriormente e imprima em forma de tabela essas informações.

**Questão 16 -** Escreva um programa fazendo o uso de struct's. Você deverá criar uma struct chamada Ponto, contendo apenas a posição x e y (inteiros) do ponto. Declare 2 pontos, leia a posição (coordenadas x e y) de cada um e calcule a distância entre eles. Apresente no final a distância entre os dois pontos.

**Questão 17 -** a) Explique a diferença entre p++; (*p)++; *(p++).

- O que quer dizer *(p+10);? Explique o que você entendeu da comparação entre ponteiros.

b) Qual o valor de y no final do programa? Tente primeiro descobrir e depois verifique no computador o resultado. A seguir, escreva um /* comentário */ em cada comando de atribuição explicando o que ele faz e o valor da variável à esquerda do '=' após sua execução.

```
int main()
{
    int y, *p, x;
    y = 0;
    p = &y;
    x = *p;
    x = 4;
    (*p)++;
    x--;
    (*p) += x;
    printf ("y = %d\n", y);
    return(0);
}
```

**Questão 18 -** Faça um programa que inverta uma string: leia a string com gets e armazene-a invertida em outra string. Use o comando for para varrer a string até o seu final.

**Questão 19 -** Escreva um programa que peça três inteiros, correspondentes a dia , mês e ano. Peça os números até conseguir valores que estejam na faixa correta (dias entre 1 e 31, mês entre 1 e 12 e ano entre 1900 e 2100). Verifique se o mês e o número de dias batem (incluindo verificação de anos bissextos). Se estiver tudo certo imprima o número que aquele dia corresponde no ano. Comente seu programa.

PS: Um ano é bissexto se for divisível por 4 e não for divisível por 100, exceto para os anos divisíveis por 400, que também são bissextos.

**Questão 20 -** Faça um programa que leia de UMA VEZ SÓ quatro palavras pelo teclado, e armazene cada palavra em uma string. Depois, concatene todas as strings lidas numa única string. Por fim apresente esta string única como resultado ao final do programa.