Universidade Federal de Uberlândia
Curso de Graduação em Gestão da Informação
1º Avaliação de Estruturas de Dados I – Prof. Daniel A. Furtado
Introdução à POO e o Tipo Abstrato de Dados Pilha – Valor: 25 pontos

Data/Hora de Início: 10/08/2021 – **10h40**
Data/Hora de Término: 10/08/2021 – **12h20**

INTRUÇÕES GERAIS

- Esta avaliação deve ser realizada individualmente;
- As soluções devem ser implementadas utilizando a linguagem C# ou Python. Questões com respostas implementadas em outras linguagens não serão consideradas;
- O arquivo correspondente à solução de cada questão (.cs ou .py) deve ser enviado pelo sistema SAAT (www.furtado.prof.ufu.br), sem compactar, **até o horário de término indicado acima**. Apenas **um arquivo deve ser enviado por questão.** Portanto, todo o código da questão (definições de classes, programa principal, etc.) deve ser colocado no mesmo arquivo.
- Questões não enviadas até a data/hora limite serão automaticamente perdidas pelo aluno (com respectiva nota 0);
- Esta prova pode ser realizada com consulta **apenas** aos materiais disponibilizados pelo professor no endereço www.daniel.prof.ufu.br e no MS Teams, incluindo slides de aula e códigos de exemplo. Porém, NÃO É PERMITIDA a consulta a outros materiais, websites, trabalhos de semestres anteriores, etc. O uso de qualquer material desse tipo será interpretado como plágio e resultará na anulação da prova;
- Todo o código deve ser indentado corretamente. Código sem a devida indentação será considerado ilegível, de difícil manutenção, e resultará em penalização de 30% sobre o valor da questão;
- As questões devem ser resolvidas com implementações eficientes;
- As implementações não devem conter qualquer conteúdo de caráter imoral, desrespeitoso, pornográfico, discurso de ódio, desacato, etc.;
- Respostas de questões enviadas por e-mail **não serão consideradas**;
- **Alunos envolvidos em qualquer tipo de plágio, total ou parcial, terão suas provas anuladas e receberão nota zero (art. 196 do Regimento Geral da UFU).**

Questão 1 (9 pontos)

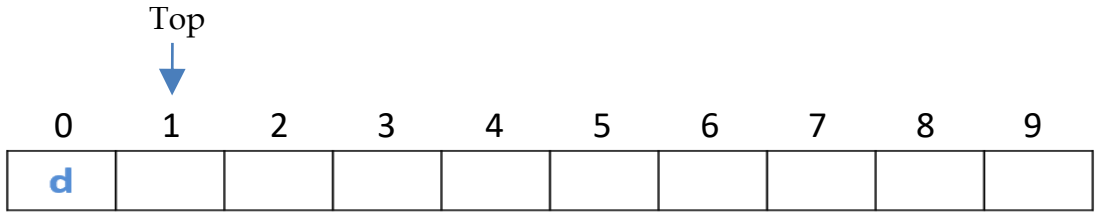
a) **(6 pontos)** Defina uma classe que permita a modelagem de um triangulo a partir de sua base e altura. A classe deve possuir atributos e métodos conforme especificação a seguir:
   - A base e a altura do triangulo devem ser modeladas como atributos do tipo *float*;
   - A classe deve ter um construtor que permita criar e inicializar novos objetos dados os valores da base e altura do novo triângulo;
   - A classe deve possuir um método para calcular e retornar a área do triangulo, que é dada pela fórmula a seguir, onde b é a base e h é a altura:

$$A = \frac{b \,.\, h}{2}$$

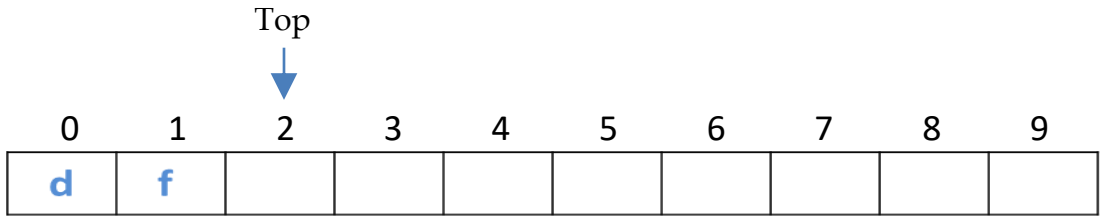
b**) (3 pontos)** Crie um programa principal que faça uso da classe anterior. O programa deve mostrar a criação de dois objetos da classe (dois triângulos), e utilizar adequadamente o método para cálculo da área (Atenção: ao terminar, envie o arquivo pelo sistema SAAT).

Questão 2 (8 pontos)

Implementar o Tipo Abstrato de Dados Pilha em ***array***, seguindo as especificações descritas a seguir, para armazenar valores do tipo **`char`** (caracteres). Utilize como base o trecho de código disponibilizado a seguir e complete adequadamente as partes indicadas nos comentários com o código adequado para desempenhar a respectiva operação. **Atenção:** a implementação deve considerar que o atributo **top** aponte para a **próxima posição livre** no topo da fila, se houver (quando a pilha estiver cheia, top poderá apontar para uma posição inexistente no vetor, o que não é um problema). Observe o diagrama a seguir:



Estado da pilha depois de executar `Push('d')`



Estado da pilha depois de executar `Push('f')`

```
public class StackInArray
{
    // Não acrescente outros atributos
    char[] items;
    int top;

    public StackInArray(int capacity)
    {
        this.items = new char[capacity];
        // complete o código aqui
    }
    public void Push(char item)
    {
        // preencha com o código adequado
    }
    public char Pop()
    {
        // preencha com o código adequado
    }
    public bool IsEmpty()
    {
        // preencha com o código adequado
    }
    public bool IsFull()
    {
        // preencha com o código adequado
    }
}
```

```
class StackInArray:

    def __init__(self, size):
        self.items = array.array('u',[' '
for i in range(size)])
        self.top = # complete aqui sem
criar outros atributos

    def Push(self, newItem):
        # preencha com o código adequado

    def Pop(self):
        # preencha com o código adequado

    def IsEmpty(self):
        # preencha com o código adequado

    def IsFull(self):
        # preencha com o código adequado
```

Atenção: ao terminar a questão anterior, envie o arquivo pelo sistema SAAT imediatamente!

Questão 3 (8 pontos)

Implementar um programa que receba uma palavra do usuário (*string*) e informe se a palavra corresponde a um **palíndromo** ou não. Uma palavra constitui um palíndromo quando ela pode ser lida, indiferentemente, da esquerda para a direita ou da direita para a esquerda. Exemplos de palíndromos: ana, osso, esse, arara, abccba, aabbccccbbaa, etc. As seguintes restrições devem ser obedecidas:

- A solução deve utilizar, **obrigatoriamente, uma pilha** (pode-se utilizar a pilha implementada na questão anterior);
- A solução não deve se basear na comparação direta de caracteres correspondentes na *string* (primeiro com o último, segundo com o penúltimo, etc.);
- A solução não deve utilizar qualquer função, método ou operador de manipulação de *strings* (exceto para encontrar a quantidade de caracteres na *string*);
- Para simplificar a solução, considere que o usuário sempre fornecerá uma *string* com um número par de caracteres;
- A *string* deve ser percorrida apenas no sentido ascendente.

Em C#, pode-se utilizar o método `Console.ReadLine()` para solicitar o texto ao usuário. Em Python, pode-se utilizar a função `input()`. O tamanho da *string* pode ser resgatado em C# por meio da propriedade **Length** da *string* (veja exemplo a seguir). Em Python, pode-se utilizar a função len(*string*).

```
string str = "Minha string";
int tamanho = str.Length;
Console.WriteLine(str[0]); // mostra o 1o caracter
```

OBS: Caso a questão 2 não tenha sido resolvida, pode-se supor, para fins de solucionar a questão 3, que a classe correspondente à pilha tenha sido previamente criada. Neste caso, deve-se utilizar adequadamente a classe e seus métodos conforme definidos no código disponibilizado para a questão 2.