LabI: Um programa “Simples”

Nesse exercício você deve rodar um programa em Python que imprima na tela a frase “oi Mamãe?”. O nome do arquivo fonte deve ser SimpleProgram.

Demonstração

O que deve aparecer na tela após a execução deve ser similar a isso:
C:\>Python SimpleProgram.py
Oi Mamãe?.

LabII: Usando “For”

Para esse exercício você vai usar um loop(For) para imprimir algumas strings e outro para simular a multiplicação de (6*3). O nome do arquivo fonte deve ser ForLoop.

Etapas

1. Faça um “For” que imprima "oi" 5 vezes.
2. Defina uma variável “soma” com valor inicial 0.
3. Use dois Loops para incrementar o valor da soma e atingir o resultado = 18(6*3).
4. Imprima no final o valor de “soma”.

Demonstração

O que deve aparecer na tela após a execução deve ser similar a isso:
oi
oi
oi
oi
oi
18

LabIII: Usando Arrays

Para esse exercício você vai criar e inicializar um vetor de inteiros e um de objetos String.

Etapas

1. Declare um vetor de inteiros de tamanho 5.
2. Use um loop while para preencher o vetor com valores acima de 10.
3. Use um For para imprimir os valores do vetor.
4. Declare um vetor de strings inicializado com Zé, João, e Tonho.
5. Use um For para imprimir os valores do vetor.
6. Mude o valor do primeiro elemento do vetor para Maria.
7. Use um For para imprimir os valores do vetor.

Demonstração

O que deve aparecer na tela após a execução deve ser similar a isso:

```
10
11
12
13
14
Zé
João
Tonho
Maria
João
Tonho
```

LabIV: Usando If

Para esse exercício você vai definir variáveis para armazenar temperaturas e imprimir, quando ou não, o clima está quente, frio, normal, ou extrema. A temperatura atual deve ser passada pelo teclado.

Etapas

1. Defina as variáveis quente=40, frio=10, e atual que recebe o valor passado. Imprima o valor de atual.

2. Use a estrutura “IF” para testar se a temperatura atual é igual a frio. Se verdadeiro, imprima "FRIO." Senão, teste para ver se a temperatura atual é igual a quente. Se for, imprima "Quente".

3. Use a estrutura “IF” para testar se a temperatura atual está entre frio e quente. Se verdadeiro, imprima "Normal". Senão, imprima “Temperatura Extrema”.

Demonstração

O que deve aparecer na tela após a execução deve ser similar a isso:
C:\>Python ProgramaIf.py
Atual tem valor <um valor>
Normal

LabV: Definindo e Usando Métodos em Python

Para esse exercício, você vai definir e usar alguns métodos em Python.

Etapas

1. Defina um método chamado “Welcome” em uma classe MetodoClass, que não tem argumentos(parâmetros) nem valor de retorno. O método deve apenas imprimir "Seja Bem Vindo!!!."

2. Faça o método main da MetodoClass chamar o Welcome ().

3. Defina outro método chamado addTwo que pega um valor inteiro e soma 2 a ele, retornando o resultado.

4. No método main da MetodoClass, defina uma variável local inteira com valor 3 e em seguida chame addTwo(i) passando ela como parâmetro. Imprima o valor retornado pelo método. Repita o passo anterior mudando o valor da variável para 19. Imprima o resultado

Demonstração

O que deve aparecer na tela após a execução deve ser similar a isso:

Seja bem-vindo
addTwo(3) é 5
addTwo(19) é 21

LabVI: A Classe MusicStore

Para esse exercício você vai implementar a primeira versão de um tipo abstrato de dados definido pelo programador de nome MusicStore. Essa Classe será usada exercícios subsequentes.

Etapas

1. Monte e implemente o MusicStore como uma classe pública com um método público de nome displayHoursOfOperation. Esse método imprime na tela o período diário de funcionamento de uma loja de música(discos).

2. Monte e implemente o TestMusicStore como uma classe pública com o método main que execute as seguintes tarefas:
    - Criar uma instância do MusicStore
    - Invocar o método displayHoursOfOperation para período diário de funcionamento

3. Compile os fontes de MusicStore e TestMusicStore.
4. Execute TestMusicStore.

Demonstração

O que deve aparecer na tela após a execução deve ser similar a isso:

Período:
Diariamente das 9:00 AM - 21:00 PM

LabVII: MusicStore com um Dono

Para esse exercício você adiciona uma variável de instância chamada "owner" e o método setOwner a MusicStore.

Etapas

1. Adicione a variável "owner" a MusicStore. O tipo deve ser String inicializado com "sem dono.".

2. Adicione o método setOwner to MusicStore. Esse método deve modificar o valor da variável "owner".

3. Modifique TestMusicStore para mudar o nome do dono da loja.

4. Compile os fontes de MusicStore e TestMusicStore.

5. Execute TestMusicStore.

Demonstração

O que deve aparecer na tela após a execução deve ser similar a isso:
Período:
Diariamente das: 9:00 AM - 21:00 PM

Roberto, Proprietário

LabVIII: MusicStore - Aberta ou Fechada?

Para esse exercício você adiciona algumas variáveis e métodos para manipular as condições de aberta ou fechada da MusicStore.

Etapas

1. Crie as variáveis openTime e closeTime para a MusicStore. O tipo de dados deve ser inteiro com valores 9 e 21 respectivamente.

2. Crie 4 métodos de acesso para as variáveis definidas: setOpen, getOpen, setClose, e getClose.

3. Crie um método de nome isOpen que retorna o valor "boleano" indicando se a loja está aberta ou fechada no momento. O método deve comparar as varáveis openTime e closeTime com o valor da hora do sistema. Você usará o método getHourInt descrito abaixo para obter o valor da hora do sistema.

4. Crie por conveniência o método getOpenClosedMessage. Ele deve retornar uma mensagem avisando quando a loja está fechada ou aberta, baseada no valor do método isOpen.

5. Então, modifique o método displayHoursOfOperation, que antes mostrava valores arbitrários para os horários de abertura e fechamento da loja, para mostar os valores especificados nas variáveis openTime e closeTime.

6. Modifique a TestMusicStore para mostrar mensagens do tipo "Estamos Abertos!" ou "Estamos Fechados!".

7. Compile os fontes de MusicStore e TestMusicStore.

8. Execute TestMusicStore.

Demonstração

O que deve aparecer na tela após a execução deve ser similar a isso:
Período:
Estamos abertos!!
Diariamente das: 9:00 AM - 21:00 PM

Roberto, Proprietário

LabIX: MusicStore – Concatenação de String

Para esse exercício você (1) vai modificar o displayHoursOfOperation para que ele use a data e a hora atuais e (2) criar um método toString para MusicStore.

Etapas

1. Modifique o método displayHoursOfOperation para que ele apresente os horários corretos de abertura e fechamento, isso é, horários consistentes com os valores armazenados nas respectivas variáveis.

2. Crie um método toString para MusicStore que concatene junto a informação pertinente para a instância corrente e retorne a String resultante.

3. Modifique a TestMusicStore para Testar/Mostrar a funcionalidade do método of toString.

4. Compile os fontes de MusicStore e TestMusicStore.

5. Execute TestMusicStore.

Demonstração

O que deve aparecer na tela após a execução deve ser similar a isso:
Período:
[Dono = Carlos , Abre = 9, Fecha = 21]

LabX: MusicStore - Adicionando Títulos

Para esse exercício adicione a capacidade de manipulação de múltiplos títulos de música(discos) , isso é, armazenar, recuperar, e Mostrar títulos.

Etapas

1. Crie uma classe chamada MusicTitle com 2 Strings, title e artist, inicializadas com "sem nome". Implemente get e set métodos de acesso para ambas as variáveis.

2. Crie a variável de instância titles do tipo MusicTitle[] para MusicStore, inicializando-a com null, e então implemente os métodos de acesso setTitles() e getTitles().

3. Crie um método de nome displayMusicTitles() para MusicStore que percorra o vetor de títulos e mostre o nome e o artista do título.

4. Modifique TestMusicStore para Testar/Mostrar a funcionalidade do método displayMusicTitles().

5. Compile os fontes.

6. Execute TestMusicStore.

Demonstração

O que deve aparecer na tela após a execução deve ser similar a isso:

Título 1:
Título: A Festa
Artista: Ivete Sangalo

Título 2:
Título: Luna Nueva
Artista: Diego Torres