


Universidade do Estado do Amazonas
Escola Superior de Tecnologia
Disciplina: Tópicos Especiais em Engenharia de Software
Professor: Jonathas Silva dos Santos
Aluno: Gabriel Alexander Farias de Lima Teixeira

Teste Prático 1

O programa:

O programa escolhido foi baseado na questão 1279 - Ano Bissexto ou Ano Não Bissexto da plataforma Beecrowd (antigo URI Online Judge), com algumas pequenas modificações. A descrição do problema é a seguinte:

Ano Bissexto ou Ano não Bissexto

Por Shahriar Manzoor  Bangladesh

Timelimit: 2

A antiga raça de Gulamatu é muito avançada no seu esquema de cálculo dos anos. Eles entendem o que é ano bissexto (ano que é divisível por 4 e não é divisível por 100, com a ressalva de que ano que são divisíveis por 400 são também anos bissextos.), E têm também alguns anos que ocorrem alguns festivais. Um deles é o festival Huluculu (acontece em anos divisíveis por 15) e o festival Bulukulu (acontece em anos divisíveis por 55 desde que também seja um ano bissexto). Dado um ano você terá de indicar quais propriedades este ano tem. Se o ano não é ano bissexto e nem ano de festival imprima a linha 'This is an ordinary year.', ou seja, que é um ano comum. A ordem de impressão das propriedades dos anos (se presente) é leap year -> huluculu -> bulukulu.

Entrada

A entrada conterá vários casos de teste. Cada caso de teste consiste de uma linha contendo um ano que nunca será menor do que 2000 (para evitar regras anteriores diferentes para anos bissextos), mas pode ter mais do que 1.000 dígitos. O final da entrada é determinado por fim de arquivo (EOF).

Saída

Para cada entrada, imprima as diferentes propriedades dos anos em diferentes linhas de acordo com a descrição anterior e os exemplos fornecidos abaixo. Uma linha em branco deve separar

cada caso de teste de saída. Note que existem quatro diferentes propriedades. Obviamente não deverá ter uma linha em branco após o último caso de teste.

O sistema foi programado usando a linguagem Python 3.9. A implementação do sistema (*bissexto.py*) foi feita da seguinte forma com a adição de algumas regras:

```
# -*- coding: utf-8 -*-
def Bissexto(ano):
    #verifica se é ordinário
    ord = 1
    #verifica se é bissexto
    bis = 0
    #string para fazer o print de acordo com o tipo de ano
    tipo = ''

    #input do ano e se verifica se é inválido
    if (ano < 2000) or (ano > 10000):
        tipo += 'INVÁLIDO'
        print(tipo)
        return

    #calcula se é bissexto
    if (ano % 4 == 0) and (not (ano % 100 == 0) or (ano % 400 == 0)):
        ord = 0
        bis = 1
        tipo += 'Bissexto'

    #calcula se é huluculu
    if (ano % 15 == 0):
        ord = 0
        if (tipo != ''):
            tipo += ', Huluculu'
        else:
            tipo += 'Huluculu'

    #calcula se é bulukulu
    if (ano % 55 == 0) and bis:
        tipo += ', Bulukulu'

    #se nenhum dos anteriores, é ordinário
    if ord:
        tipo += 'Ordinário'

    print(tipo)
```

O sistema começa recebendo a variável **ano**, o input do ano que se quer a classificação. Ambas **bis** e **ord** são variáveis que servem como booleanos. A variável **tipo** recebe os tipos de ano relacionados com o ano digitado. O primeiro if verifica se é um ano válido (igual ou maior que 2000 e menor ou igual a 10000), e caso não

seja, **ord** vira 0 (Falso) e **tipo** recebe "INVÁLIDO", e logo depois retorna a função. Os 3 *ifs* seguintes calculam se **ano** é um ano bissexto, e se for, **bis** recebe 1 e permite que o cálculo de ano com festival Bulukulu aconteça; o segundo *if* verifica se é ano de Huluculu, e se for, será impresso logo após a mensagem de ano bissexto; o terceiro verifica se é bissexto e divisível por 55, e assim pode ser um ano Bulukulu; se não for nenhum dos outros, **ord** indica que é um ano ordinário. No fim, a variável **tipo** é escrita na tela com a saída esperada.

Técnica Funcional - Partição de Equivalência

As modificações no programa: Todos os anos menores que 2000 serão considerados **inválidos**, assim como todos aqueles maiores que 10000. O sistema se baseia nas entradas a partir de 2000, onde são contabilizados os novos anos que estão de acordo com as novas regras de definição de anos bissextos. O sistema chega a ter 7 tipos de saída: inválida, ano bissexto, ano de huluculu, ano bissexto e de bulukulu, bissexto e huluculu, bissexto e de huluculu e de bulukulu, e por fim, ano ordinário. Não é possível ter ano de bulukulu por si só, visto que é necessário que também seja um ano bissexto.

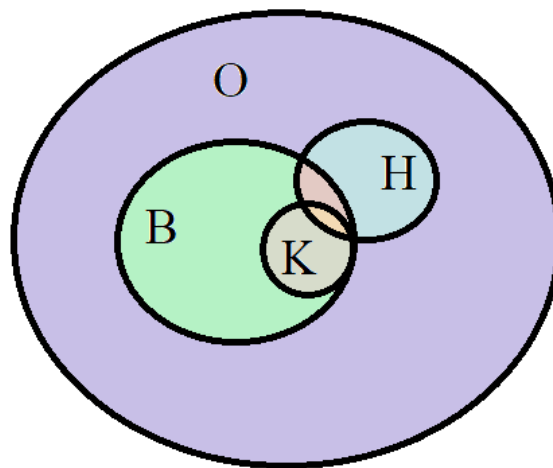
A partir dessas informações, usando o intervalo definido, é possível identificar 3 classes de entrada: 2 inválidas (valor menor que 2000 e maior que 10000) e 1 válida (entre 2000 e 10000). Dentre as saídas válidas, existem os 7 tipos de saída mencionadas anteriormente.

ROTEIRO DE TESTES 1

| CT_ID | Entradas (Anos) | Saídas Esperadas |
|-------|-----------------|------------------------------|
| CT_01 | 1500 | INVÁLIDO |
| CT_02 | 20000 | INVÁLIDO |
| CT_03 | 2015 | Ordinário |
| CT_04 | 2000 | Bissexto |
| CT_05 | 4515 | Huluculu |
| CT_06 | 2420 | Bissexto, Bulukulu |
| CT_07 | 3600 | Bissexto, Huluculu |
| CT_08 | 2640 | Bissexto, Huluculu, Bulukulu |

Técnica funcional - Análise do Valor Limite

Ao observar as definições de entrada, o intervalo gerado entre 2000 e 10000 possui dois limites: mínimo (2000) e máximo (10000). Os valores dentro do conjunto do intervalo, estão em classes de equivalência diferentes, porém não possuem um limite visível ou completamente definido; por se tratarem de valores matemáticos, existem fórmulas que identificam os valores de determinado conjunto, porém a distância entre um valor e outro do mesmo conjunto é grande. Portanto, não serão considerados os limites das saídas válidas, devido à distância entre um valor e outro do mesmo conjunto. Uma imagem que diferencia os conjuntos de valores foi feita para demonstração dos mesmos:



Onde:

- **O** é o conjunto dos anos Ordinários;
- **B** é o conjunto dos anos Bissextos;
- **H** é o conjunto dos anos Huluculu;
- **K** é o conjunto dos anos Bulukulu;
- Há a interseção dos anos que são Bissextos e Huluculu;
- Há a interseção dos anos que são Bissextos, Huluculu e Bulukulu;
- Os anos Bulukulu são anos bissextos, portanto estão dentro do conjunto dos anos Bissextos.

Com estes dados em mente, os testes considerando os valores limite ficaram da seguinte forma:

| Limite Mínimo | Limite Máximo |
|--------------------|----------------------|
| {1999, 2000, 2001} | {9999, 10000, 10001} |

ROTEIRO DE TESTES 1

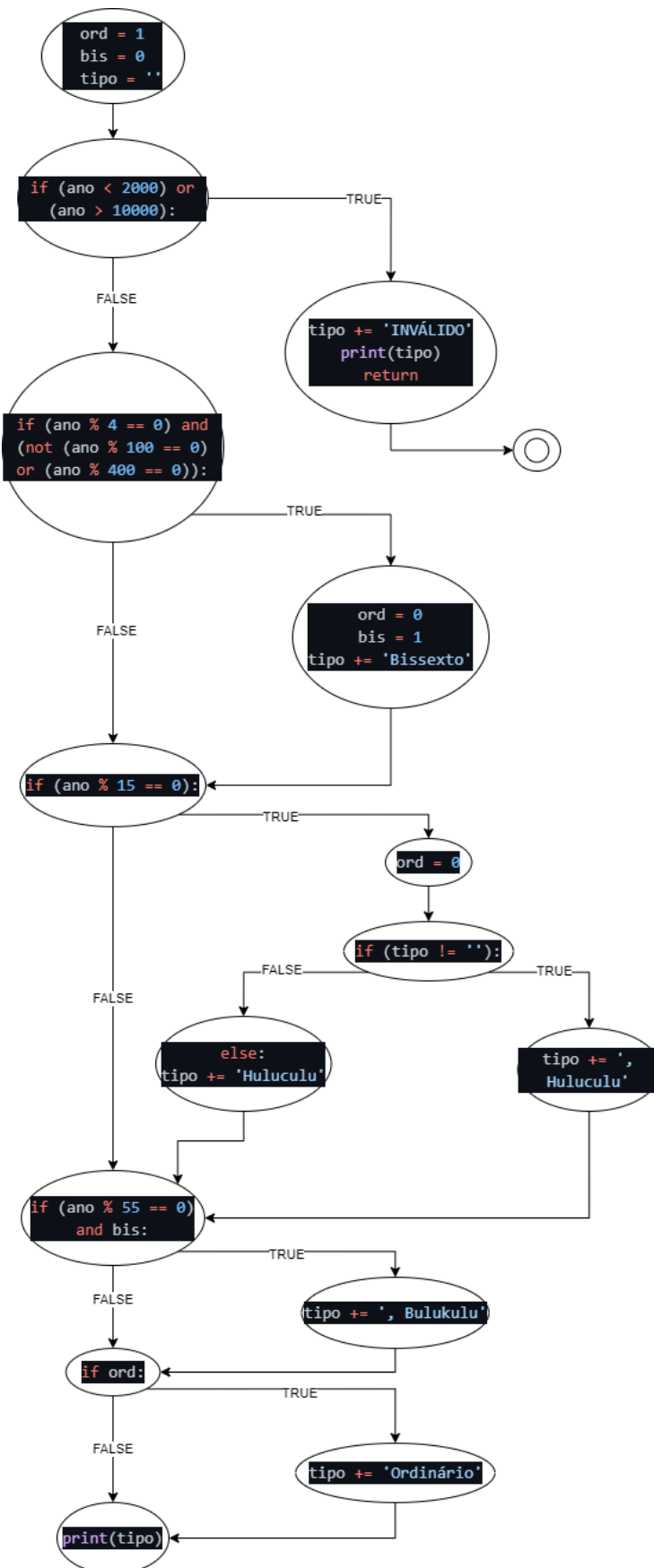
| CT_ID | Entradas (Anos) | Saídas Esperadas |
|-------|-----------------|------------------------------|
| CT_01 | 1999 | INVÁLIDO |
| CT_02 | 2000 | Bissexto |
| CT_03 | 2001 | Ordinário |
| CT_04 | 9999 | Ordinário |
| CT_05 | 10000 | Bissexto |
| CT_06 | 10001 | INVÁLIDO |
| CT_07 | 2015 | Ordinário |
| CT_08 | 4515 | Huluculu |
| CT_09 | 2420 | Bissexto, Bulukulu |
| CT_10 | 3600 | Bissexto, Huluculu |
| CT_11 | 2640 | Bissexto, Huluculu, Bulukulu |

Refinando os valores, o roteiro se estabelece da seguinte forma:

ROTEIRO DE TESTES 1

| CT_ID | Entradas (Anos) | Saídas Esperadas |
|-------|-----------------|------------------------------|
| CT_01 | 1999 | INVÁLIDO |
| CT_02 | 2000 | Bissexto |
| CT_03 | 2001 | Ordinário |
| CT_04 | 10000 | Bissexto |
| CT_05 | 10001 | INVÁLIDO |
| CT_06 | 4515 | Huluculu |
| CT_07 | 2420 | Bissexto, Bulukulu |
| CT_08 | 3600 | Bissexto, Huluculu |
| CT_09 | 2640 | Bissexto, Huluculu, Bulukulu |

Técnica Estrutural - Grafo de Fluxo de Controle (CFG)



Para melhor visualização, seguem os links do Drive (que referencia o Diagrams.net), e da imagem independente:

- [Grafo de Controle de Fluxo - Drive](#)
- [Grafo de Controle de Fluxo - PNG](#)

Casos de teste - Implementação

Foi instalado o **pipenv** para ser o ambiente virtual e de gerenciamento de pacotes do repositório, e dentro foram utilizados os pacotes **pytest** e **pytest-cov** para realizar os testes e verificar a cobertura dos códigos. O arquivo *test_bissexto.py* é o responsável por implementar os casos de teste do arquivo *bissexto.py*.

```
# -*- coding: utf-8 -*-
import py_compile
import unittest
import bissexto

class testBissexto (unittest.TestCase):

    def test_ct01(self):
        ano = 1999
        self.assertEqual(bissexto.Bissexto(ano), "INVÁLIDO")

    def test_ct02(self):
        ano = 2000
        self.assertEqual(bissexto.Bissexto(ano), "Bissexto")

    def test_ct03(self):
        ano = 2001
        self.assertEqual(bissexto.Bissexto(ano), "Ordinário")

    def test_ct04(self):
        ano = 10000
        self.assertEqual(bissexto.Bissexto(ano), "Bissexto")

    def test_ct05(self):
        ano = 10001
        self.assertEqual(bissexto.Bissexto(ano), "INVÁLIDO")

    def test_ct06(self):
        ano = 4515
        self.assertEqual(bissexto.Bissexto(ano), "Huluculu")

    def test_ct07(self):
```

```

        ano = 2420
        self.assertEqual(bissexto.Bissexto(ano), "Bissexto, Bulukulu")

    def test_ct08(self):
        ano = 3600
        self.assertEqual(bissexto.Bissexto(ano), "Bissexto, Huluculu")

    def test_ct09(self):
        ano = 2640
        self.assertEqual(bissexto.Bissexto(ano), "Bissexto, Huluculu,
Bulukulu")

if __name__ == '__main__':
    unittest.main()

```

Ao rodar o arquivo de teste no terminal usando **pytest** e **pytest-cov**, o arquivo é finalizado com 100% de cobertura e todos os 9 testes tiveram sua saída como “sucesso”.

```

(topicos-trabalho-1) galaxander@DESKTOP-07EP4NV:~/topicos-trabalho-1
$ pytest --cov=bissexto
===== test session starts =====
platform linux -- Python 3.9.12, pytest-7.1.1, pluggy-1.0.0
rootdir: /home/galaxander/topicos-trabalho-1
plugins: cov-3.0.0
collected 9 items

test_bissexto.py ..... [100%]

----- coverage: platform linux, python 3.9.12-final-0 -----
--
Name           Stmts   Miss  Cover
-----
bissexto.py      21      0   100%
-----
TOTAL            21      0   100%

```

Conclusões

A partir destes resultados, é possível concluir que os testes foram realizados com sucesso, a cobertura do código foi de 100% e todos os casos de testes estavam de acordo com as saídas do código principal.

O repositório no GitHub pode ser encontrado no link:

- [Tópicos em Engenharia de Software - Trabalho Prático 1 - GitHub](#)