Universidade do Estado do Amazonas
Escola Superior de Tecnologia
Data: 5 de Fevereiro de 2018
Professora: Elloá B. Guedes
Curso: Jornada Python – Semana Pedagógica IFAM 2018.1

Conhecendo a Plataforma Run.Codes
Datas Gregorianas

1 Conhecendo a plataforma Run.Codes

Para praticarmos um pouco a linguagem Python e aprender maneiras de estimular os alunos a codificarem nesta linguagem de programação vamos utilizar a plataforma *Run.Codes*.

O primeiro passo a ser realizado é o cadastro na plataforma. Acesse o site run.codes e, utilizando o seu **nome completo**, **e-mail institucional** e optando pela "Universidade do Estado do Amazonas". Após esta etapa, procure a disciplina **Linguagem de Programação 1** (ESTECP001) e cadastre-se na sua turma "Jornada Python", com o código de matrícula BEQC.

Todos os alunos devem obrigatoriamente se cadastrarem até o dia **07/02/2018**. O não cadastro, o cadastro em turma incorreta ou a não realização dos exercícios nos prazos estabelecidos culminará em prejuízo ao aprendizado.

Para quem não conhece, o run.codes é uma plataforma automatizada para testes de entrada e saída. Cada aluno submete o seu código escrito na linguagem Python e este código é submetido a um conjunto de testes, previamente escritos e cadastrados pelos professores. A nota do aluno é individual e obtida de maneira automática, correspondendo ao percentual de acertos nos testes. Por exemplo, se há 15 casos de testes cadastrados e o aluno acertou 12, a nota obtida é 8.

A partir do momento em que o exercício inicia até o momento do seu encerramento, o aluno pode submeter o código para a plataforma quantas vezes quiser, sem que isso afete a nota final. Por exemplo, se um aluno submeteu 10 vezes e acertou 12/15 casos de teste, a nota será a mesma de um outro aluno que acertou 12/15 mas que submeteu 300 vezes.

Uma outra regra a ser considerada na plataforma é que a última versão do código é sempre a que será considerada. Imagine que o aluno João Última Hora está enlouquecidamente programando o exercício e já tem 14/15 casos corretos mas, nos segundos finais do prazo limite submete alterações em seu código e cai para 8/15 acertos. Se o sistema encerrar, a versão 8/15 será a considerada para avaliação. Se João Última Hora tivesse aproveitado melhor o tempo e começado a resolver o exercício desde o momento em queficou disponível na plataforma, não teria passado esse sufoco e ficado com uma nota final tão ruim. Todos podemos aprender com o drama de João Última Hora e evitar tais problemas.

Algumas dicas finais para você ter um bom desempenho nos problemas práticos são:

1. Considere que seu programa recebe uma entrada de cada vez;
2. Efetue testes em seu programa antes de submetê-lo ao run.codes. É uma forma simples de conhecer como seu programa se comporta e uma oportunidade de acertar mais testes logo de primeira;
3. Aproveite o tempo;
4. Há boatos de que você não deve deixar seu computador saber quando você está com pressa!

O problema a ser apresentado a seguir consiste em um **treinamento** para a utilização da plataforma e não será considerado para fins de avaliação. Aproveite a oportunidade para explorar a plataforma e praticar um pouco da linguagem Python.

2 Apresentação do Problema

O calendário gregoriano é um calendário de origem européia, utilizado oficialmente pela maioria dos países. Como convenção e por praticidade o calendário gregoriano é adotado para demarcar o ano civil no mundo inteiro, facilitando o relacionamento entre as nações.

Uma data no calendário gregoriano é composta por várias informações, incluindo: um número de dois dígitos correspondente ao dia, um número de dois dígitos correspondente ao mês e um número de quatro dígitos correspondentes ao ano. Uma data é considerada válida no calendário gregoriano se respeita as regras deste tipo de calendário.

O objetivo deste exercício é construir um programa capaz de verificar se as entradas fornecidas representam datas válidas ou inválidas no calendário gregoriano. Na construção do seu programa você deve usar apenas os conceitos aprendidos em sala de aula. Respostas que utilizem bibliotecas prontas não serão consideradas.

1. **Entrada**. Caracteres na forma DD/MM/AAAA;
2. **Saída**. True, caso a data fornecida seja válida; False, em caso contrário.

3 Exemplos de Entradas e Saídas

4 Observações Importantes

- Lembre-se, a entrada de dados é feita via `input` e a saída via `print`;
- Não precisa mostrar mensagem para entrada de dados. A saída deve ser apenas a conclusão, sem qualquer mensagem adicional;

Entrada	**Saída**
30/03/2015	True
01/01/1999	True
28/02/2000	True
30/04/2050	True
99/12/2005	False
28/75/9842	False
31/02/2015	False

- A cada execução do programa será fornecida apenas uma entrada, cujo resultado deve ser exibido ao final do processamento;
- Todas as entradas a serem testadas no seu programa são bem formatadas, isto é, todas elas são no formato DD/MM/AAAA. Nenhuma entrada em formato diferente será fornecida, fique tranquilo(a) quanto a isto!
- Na construção do seu programa você deve usar apenas os conceitos aprendidos em sala de aula. Respostas que utilizem bibliotecas prontas não serão consideradas;
- Em caso de plágio, todos os envolvidos receberão nota zero!
- Na execução do seu programa no *run.codes*, existem casos de testes que vão além dos exemplos mostrados a seguir. Esses casos de teste não serão revelados. Pense em exemplos de entradas e saídas que podem acontecer e melhore o seu código para capturá-las.

5 Prazos Importantes

- **Início**. 06/02/2018 às 8h (horário do servidor)
- **Encerramento**. 08/02/2017 às 23h55min (horário do servidor)