Unidade 2: Iniciação à programação em Python — **Aplicação: Ciclos e Testes**

Nesta Aplicação da Unidade 2, propõe-se que utilize as noções trabalhadas nas lições sobre funções condicionais, bem como ciclo limitados e não limitados.

Objetivos:

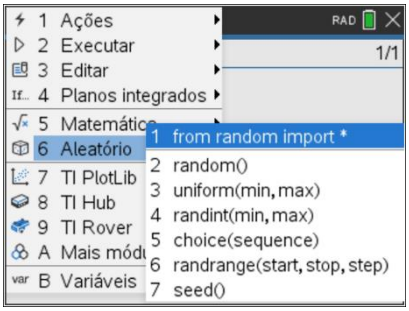
- Utilizar os ciclos **While** e **For** para implementar algoritmos para a resolução de problemas de probabilidades e de estatística.

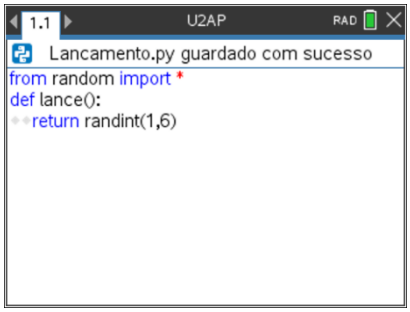
Nesta aplicação iremos elaborar um programa que permita:

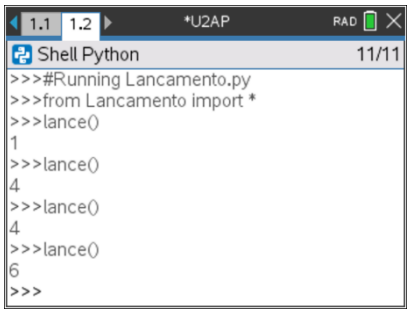
- obter um número aleatório criando, para tal, uma função que designaremos **lançamento**.
- utilizar esta função num outro programa de forma a determinar o número de lançamentos necessários para obter uma soma de 12 ao lançar 2 dados cúbicos perfeitamente equilibrados e numerados de 1 a 6.
- ao lançar um único dado de 6 faces, se obtenha o número de vezes que cada face saiu, permitindo-se desta forma calcular a frequência relativa e comparar com a probabilidade teórica.

Lançamento de um Dado

- Note que o recurso a funções da TI-Nspire CX com números aleatórios requer, na aplicação TI-Python, a ativação do módulo aleatório.
- Abra um novo documento da TI-Nspire CX II T.
- Crie uma nova página com o editor de programas da aplicação Python, designe o programa por "Lancamento" e de seguida pressione enter para validar.
- Ative o módulo aleatório pressionando a tecla menu e selecionando de seguida a opção **6: Aleatório** e depois a opção **1: from random import***.
- Defina, agora, a função **lance()** que permita obter um número inteiro aleatório entre 1 e 6, usando a função aleatória **randint()** da TI-Nspire CX.
- Teste a função **lance()**, depois de verificar e guardar o programa.
- Utilize a tecla para cima, ▲, para deslocar o cursor para cima e executar novamente a função ou utiliza a tecla var, ou ainda, escreva o nome da função.





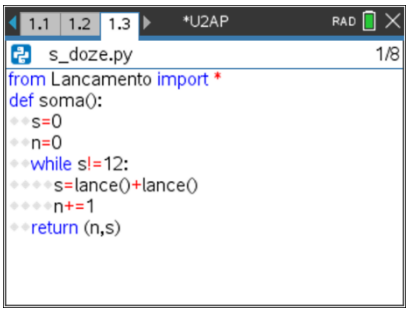


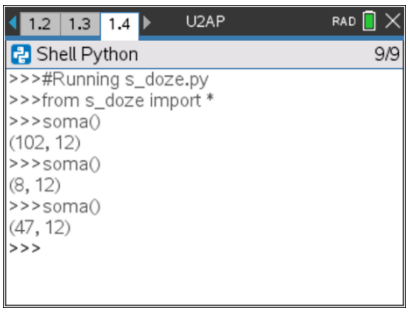
Número de lançamentos necessários.

Lance dois dados de 6 faces, cubos perfeitamente equilibrados e com as faces numeradas de 1 a 6, e adicione os dois resultados obtidos. Escreva outro programa para simular os lançamentos desses dois dados e encontrar o número n de lançamentos realizados até obter a soma 12.

Existem muitas soluções possíveis, mas a primeira que vem à mente é reutilizar a função anterior.

- Utilize-se a variável **s** para guardar a soma das faces em cada lançamento e a variável **n** para guardar o número de lançamentos necessários até a soma ser 12.
- Na linguagem Python o símbolo de diferente, ≠, é representado por **!=** . Este símbolo pode ser obtido clicando na tecla menu, seguido da opção **4: Planos integrados** e por fim a opção **3: Ops**, ou ainda usando o atalho ctrl e = com o teclado da unidade portátil.
- Construa a função **soma()**, numa nova página de editor de Python ou dentro do programa **Lancamento,** recorrendo a um ciclo **While** e à função **lance()**. Certifique-se que ativado o programa **Lancamento** e que é respeitada a indentação e, em seguida, execute-a.
- O resultado é um par ordenado em que o primeiro elemento indica o número de lançamentos necessários até sair soma 12, e o segundo elemento indica o valor da soma atingido, neste caso 12.

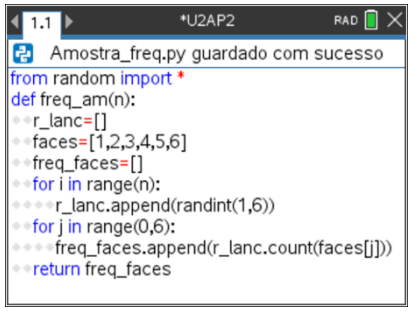




Amostragens e frequências

Acabamos de observar, no exemplo anterior, que o número de lançamentos necessários para se obter a soma 12 é variável. Podemos, portanto, proceder ao cálculo da frequência relativa da soma 12 para uma dada amostragem, que, para um grande número de simulações, deve tender para a probabilidade teórica.

- Adicione uma nova página com o editor de Python, designe o programa por **Amostra_freq**.
- Use uma função, cujo argumento será o número de simulações do lançamento do dado e que poderemos designar por **n**, para obter as frequências absoluta de cada valor numérico da face, de 1 a 6.
- Nesta função, o resultado de cada lançamento deverá ser armazenado numa lista **r_lanc**, previamente inicializada como vazia pela instrução **r_lanc = [ ]** .
- Também as faces do cubo, os números de 1 a 6, devem ser guardados inicialmente numa lista **faces**, colocando a instrução **faces=[1, 2, 3, 4, 5, 6]**.



- De seguida, usando um ciclo **FOR** são realizadas as **n** simulações do lançamento de um dado cúbico numerado de 1 a 6, sendo guardados todos resultados na lista **r_lanc**.
- Por fim, e através novamente de um ciclo **FOR**, é criada uma instrução que para cada um dos valores da variável (1 a 6) conta quantas ocorrências na lista **r_lanc** são iguais a esse valor. Essa contagem deverá ser guardada numa nova lista, **freq_faces**, que deve ser criada no início da função como uma lista vazia.
- Pressione simultaneamente as teclas ctrl e B para verificar a sintaxe e guardar o programa. Não se esqueça também de usar o atalho ctrl e S para guardar o documento.
- Execute o programa, clicando simultaneamente nas teclas ctrl e R. Abrir-se-á uma nova página com o interpretador de Python (Shell) onde foi executado o programa. Por fim execute a função **freq_am()**, colocando como seu argumento o número de simulações pretendidas, e observará a lista das frequências absolutas.
- Opcionalmente poderá adaptar o programa de forma a apresentar a frequência absoluta de um só dos valores da variável e/ou apresentar as frequências relativas. Ou então, por exemplo, tornar visível os resultados de todas as **n** simulações, que poderá ter alguma vantagem para a verificação com poucas simulações.