01

Estrutura de dados: List

Transcrição

Atualmente, já dizemos ao usuário em que posição a letra que ele chutou está na palavra secreta, caso a letra exista na palavra. Mas em um jogo real de forca, o jogador vê quantas letras há na palavra secreta. Algo como:

```
Qual letra? _ _ _ _ _ _
```

E se ele encontrar alguma letra, a mesma tem a sua lacuna preenchida. Ao digitar a letra **"a"**, ficaria:

```
_ a _ a _ a
```

Muito mais intuitivo, não? Vamos implementar essa funcionalidade.

Conhecendo uma nova estrutura de dados: lista

Para exibir as letras dessa forma, precisamos guardar os chutes certos do usuário, mas como fazer isso?

Para tal, o Python nos oferece uma estrutura de dados que nos permite guardar valores. Essa estrutura é a **lista**. Para criar uma lista, utilizamos colchetes (`[]`):

```
>>> valores = []
>>> type(valores)
<class 'list'>
```

Assim como a string, `list` também é uma sequência de dados. Então podemos ver a sua documentação (https://docs.python.org/3.6/library/stdtypes.html#sequence-types-list-tuple-range). Nela, podemos ver o que podemos fazer com uma lista, podemos verificar o seu valor mínimo com o `min` e o seu valor máximo com o `max` . Mas a nossa lista ainda tá vazia, certo? Podemos, já no momento de sua inicialização, passar valores para guardar nessa lista:

```
>>> valores = [0,1,2,3,4]
```

Agora podemos verificar seu menor e seu maior valor:

```
>>> valores = [0,1,2,3,4]
>>> min(valores)
0
>>> max(valores)
4
```

Para acessar um valor específico, podemos acessá-lo através do seu **índice**. O primeiro elemento da lista possui o índice **0**, o segundo possui o índice **1** e assim por diante:

```
>>> valores = [0,1,2,3,4]
>>> valores[2]
2
```

Podemos também saber o tamanho da lista com o `len` e verificar se determinado valor está guardado nela:

```
>>> valores = [0,1,2,3,4]
>>> 0 in valores
True
>>> 8 in valores
False
```

Além disso, existem funções específicas da lista, que podem ser vistas aqui (https://docs.python.org/3.6/library/stdtypes.html#mutable-sequence-types). Podemos adicionar elementos ao final da lista com o `append`, exibir e remover um elemento de determinada posição com o `pop`, entre diversas outras funcionalidades.

Agora que sabemos como guardar valores em uma lista, podemos voltar ao nosso jogo e guardar os acertos do usuário. Faremos isso no próximo vídeo :)