

# Algoritmos para Análise de Sequências Biológicas

## Ficha 3

# Objetivo

- Dogma central
- Tradução
- Reading frames
- Listas

# Dogma Central

## Dogma central

DNA => RNA => Aminoácidos

## DNA => RNA

Transformar T em U

## DNA => Aminoácidos

- Agrupar 3 a 3
- Usar a tabela de tradução para transformar o codão no aminoácido correspondente

## Proteínas

- Começa por um codão **M**
- Acaba por um codão de **Stop**
- Não pode conter codões de **Stop**

# Tabela de Conversão

3D Molecular Designs		The Standard Genetic Code				Center for Biomolecular Modeling	
	U	C	A	G			
U	UUU → Phe <b>F</b>	UCU → Ser <b>S</b>	UAU → Tyr <b>Y</b>	UGU → Cys <b>C</b>	U		
	UUC → Phe <b>F</b>	UCC → Ser <b>S</b>	UAC → Tyr <b>Y</b>	UGC → Cys <b>C</b>	C		
	UUA → Leu <b>L</b>	UCA → Ser <b>S</b>	UAA → Stop	UGA → Stop	A		
	UUG → Leu <b>L</b>	UCG → Ser <b>S</b>	UAG → Stop	UGG → Trp <b>W</b>	G		
C	CUU → Leu <b>L</b>	CCU → Pro <b>P</b>	CAU → His <b>H</b>	CGU → Arg <b>R</b>	U		
	CUC → Leu <b>L</b>	CCC → Pro <b>P</b>	CAC → His <b>H</b>	CGC → Arg <b>R</b>	C		
	CUA → Leu <b>L</b>	CCA → Pro <b>P</b>	CAA → Gln <b>Q</b>	CGA → Arg <b>R</b>	A		
	CUG → Leu <b>L</b>	CCG → Pro <b>P</b>	CAG → Gln <b>Q</b>	CGG → Arg <b>R</b>	G		
A	AUU → Ile <b>I</b>	ACU → Thr <b>T</b>	AAU → Asn <b>N</b>	AGU → Ser <b>S</b>	U		
	AUC → Ile <b>I</b>	ACC → Thr <b>T</b>	AAC → Asn <b>N</b>	AGC → Ser <b>S</b>	C		
	AUA → Ile <b>I</b>	ACA → Thr <b>T</b>	AAA → Lys <b>K</b>	AGA → Arg <b>R</b>	A		
	AUG → Met <b>M</b>	ACG → Thr <b>T</b>	AAG → Lys <b>K</b>	AGG → Arg <b>R</b>	G		
G	GUU → Val <b>V</b>	GCU → Ala <b>A</b>	GAU → Asp <b>D</b>	GGU → Gly <b>G</b>	U		
	GUC → Val <b>V</b>	GCC → Ala <b>A</b>	GAC → Asp <b>D</b>	GGC → Gly <b>G</b>	C		
	GUA → Val <b>V</b>	GCA → Ala <b>A</b>	GAA → Glu <b>E</b>	GGA → Gly <b>G</b>	A		
	GUG → Val <b>V</b>	GCG → Ala <b>A</b>	GAG → Glu <b>E</b>	GGG → Gly <b>G</b>	G		

  

<span style="border: 1px solid green; padding: 2px;"> </span> translation start codon	<span style="background-color: yellow; padding: 2px;"> </span> hydrophobic amino acids	<span style="background-color: red; padding: 2px;"> </span> negatively charged amino acids	<span style="background-color: lightgreen; padding: 2px;"> </span> cysteine
<span style="border: 1px solid red; padding: 2px;"> </span> translation stop codon	<span style="background-color: white; padding: 2px;"> </span> hydrophilic non-charged amino acids	<span style="background-color: lightblue; padding: 2px;"> </span> positively charged amino acids	

Figure 1: Tabela de conversão de codão para aminoácido

# Listas

## Características

- Objeto com vários métodos

## Métodos

- `append` Permite adicionar um elemento no fim
- `count` conta o nº de ocorrências de um elemento
- `index` Devolve o 1º índice de um elemento ou -1
- `pop` Remove um elemento por índice (por omissão o último)

## Operadores

- `+` Concatenação
- `in` verifica se uma string está contida noutra
- `[]` índices sobre strings

# Listas

## Percorrer listas

```
for elt in lista:  
    fazer algo com o elt
```

## Percorrer listas por índice

```
for idx in range(len(lista)):  
    fazer algo com lista[idx]
```

## Percorrer listas por índice e valor

```
for idx, elt in enumerate(lista):  
    podemos usar o índice idx e o valor elt
```

# Exemplos

```
>>> seq = "ACCTTGCA"
>>> l = []
>>> for x in seq:
...     l.append(x)
>>> l
['A', 'C', 'C', 'T', 'T', 'G', 'C', 'A']
>>> for i in range(0, len(seq), 3): print(seq[i : i + 3])
...
ACC
TTG
CA
```

# Exercícios

## Sugestões

- Os resultados devem ser sempre em maiúsculas
- Reutilize funções
- Crie funções auxiliares onde faça sentido
- Conceito avançado e poderoso: **listas por compreensão**



# Exercícios

## Exercícios

- 1 Escreva a função `get_codons(dna)` que recebe uma sequência de DNA e devolve uma lista de codões
- 2 Escreva a função `codon_to_amino(codons)` que recebe a lista de codões e devolve a sequência de aminoácidos (use `_` para o codão de **Stop**)
- 3 Escreva a função `get_prots(amino)` que recebe uma sequência de aminoácidos e devolve uma lista de possíveis proteínas
- 4 Escreva a função `get_orfs(dna)` que recebe uma sequência de DNA e devolve uma lista com as seis ORFs
- 5 Escreva a função `get_all_prots(dna)` que recebe uma sequência de DNA e devolve a lista com todas as proteínas

# Exemplos de listas por compreensão

## Carateres que não são bases de DNA

```
[x for x in dna.upper() if x not in 'ACGT']
```

## Tradução numa linha

```
[codon_to_amino(cod) for cod in get_codons(dna)]
```

## Todas as proteínas

```
[prot for orf in get_orfs(dna) for prot in get_prots(orf)]
```