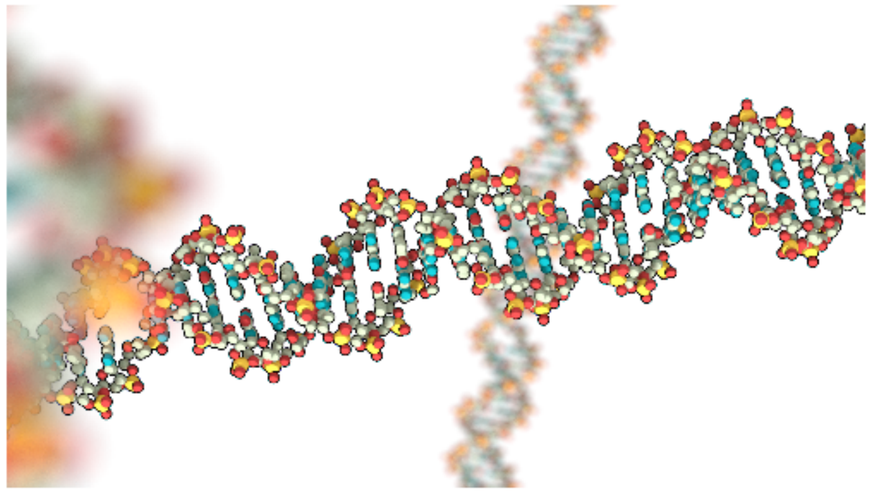
BIOS1100 H17 uke 11

Lex Nederbragt



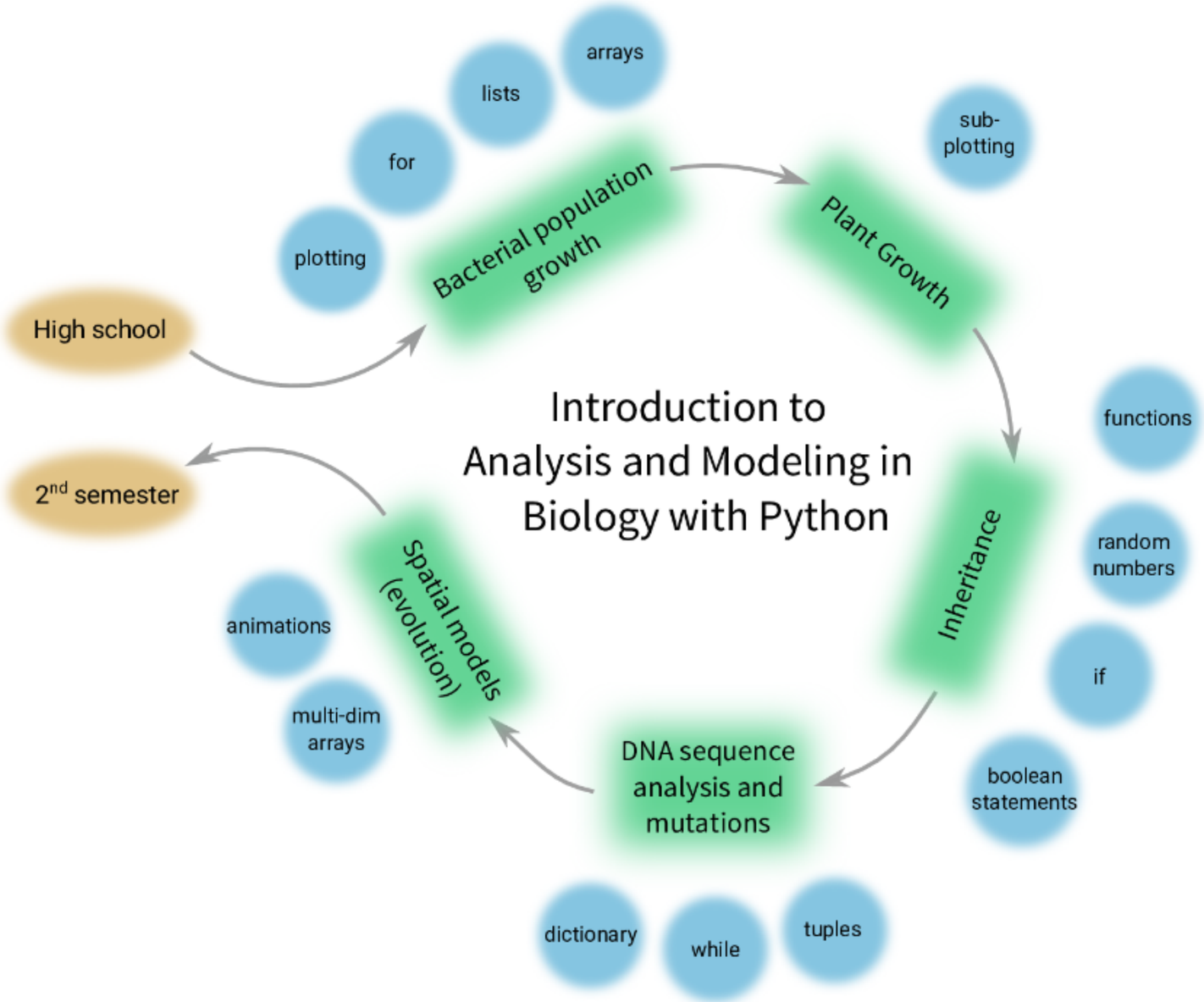
Ukens forelesning

- noen praktiske ting
- begrepsforståelse kapittel forrige uke
- nytt stoff denne uken
- utvalgte øvelser

Noen praktiske ting

- prøveeksamen
- cheat-sheet
- midtveisevaluering

Undervisningsplan



Læringsmål denne uke

- Kunne bruke `for` og `while`-løkker
- Kunne oversette `for`-løkker til `while`-løkker og omvendt
- Kunne bruke `and` og `or` i `if` eller `while` konstrukter
- kunne gjøre transkripsjons og translasjon med Python

Hva gjør denne koden?

```
my_number = 4

if my_number > 2 and my_number < 5:
    print("my_number is between 2 and 5!")
else:
    print("my_number is not between 2 and 5!")
```

Hva gjør denne koden?

```
my_number = 6

if my_number > 2 and my_number < 5:
    print("my_number is between 2 and 5!")
else:
    print("my_number is not between 2 and 5!")
```

Hva gjør denne koden?

```
my_number = 6

if my_number < 2 or my_number > 5:
    print("my_number is smaller than 2, or larger than 5!")
else:
    print("my_number is something else!")
```

and **og** or

Sett deg:

`and` **`og`** `or`

Sett deg:

- de som har på seg noe hvit og de som har på seg dongeribukse

`and` **`og`** `or`

Sett deg:

- de som har på seg noe hvit og de som har på seg dongeribukse
- de som har på seg noe hvit og dongeribukse

`and` **`og`** `or`

Sett deg:

- de som har på seg noe hvit og de som har på seg dongeribukse
- de som har på seg noe hvit og dongeribukse
- **de som har på seg noe hvit eller dongeribukse**

and **og** or

Sit down:

- those wearing something white and those wearing jeans
- those wearing something white AND jeans
- those wearing something white OR jeans

and **og** or

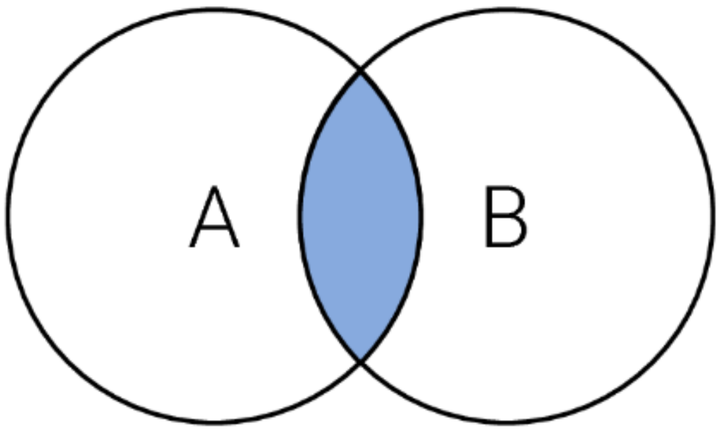
Sit down:

- those wearing something white and those wearing jeans
- those wearing something white AND jeans
- those wearing something white OR jeans

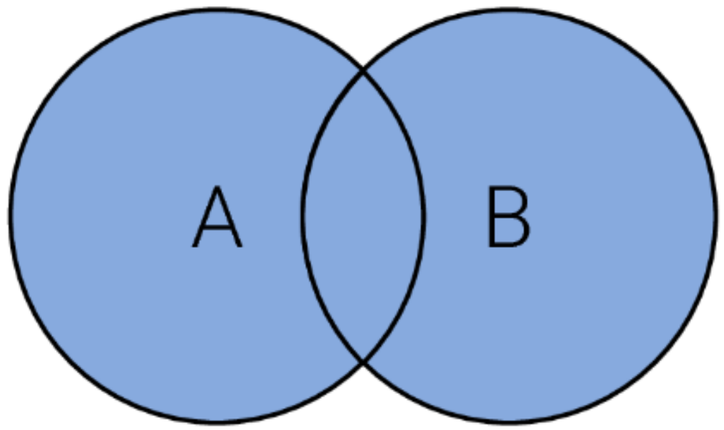
OR is MORE

A	B	A and B
True	True	True
True	False	False
False	True	False
False	False	False

A	B	A or B
True	True	True
True	False	True
False	True	True
False	False	False

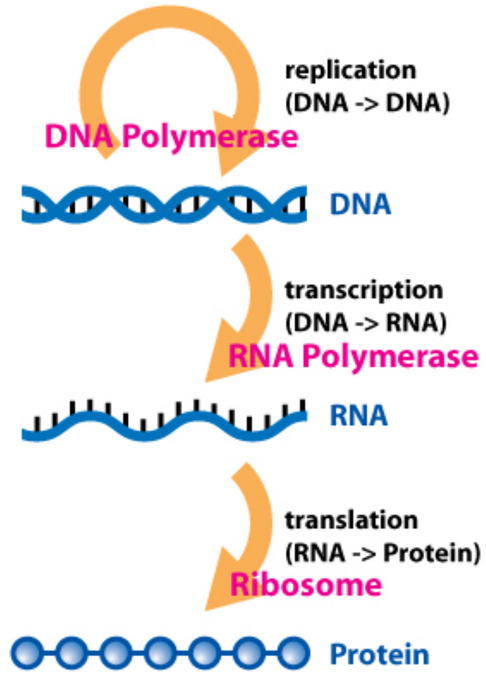


A and B



A or B

Central dogma



Neste uke i BIOS1110

Denne uken

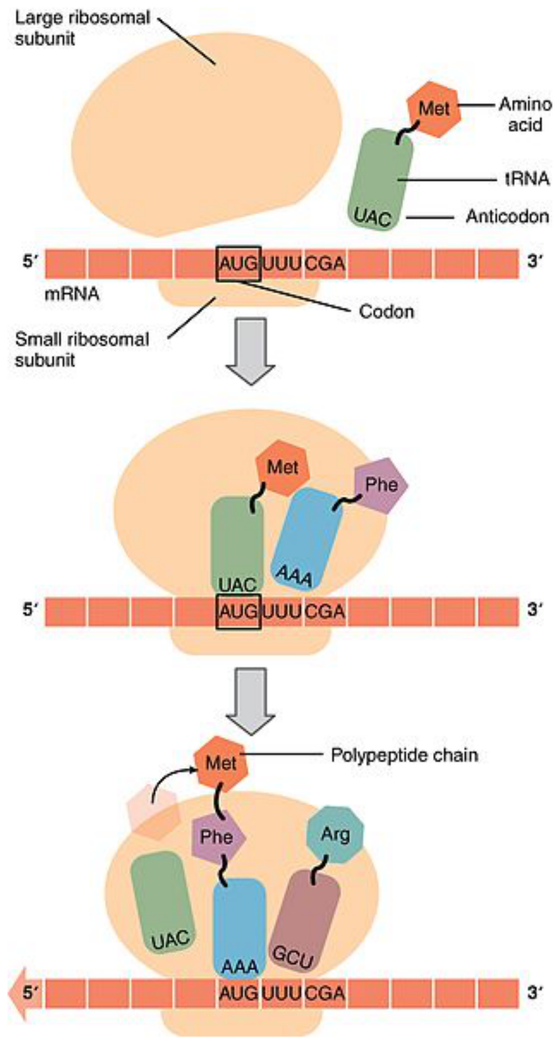
'Gjøre' stegene i central dogma med python

- Transcription of DNA to mRNA
- Translation of mRNA to Protein

Transcription of DNA to mRNA

```
E_coli_dna = "AGCTTTTCATTCTGACTGCAACGGGCAATATGTCTCTGTGTGGATTAAAA"
E_coli_rna = e_coli_dna.replace("T", "U")

print(E_coli_rna)
```

Translation of mRNA to Protein



Translation of mRNA to Protein

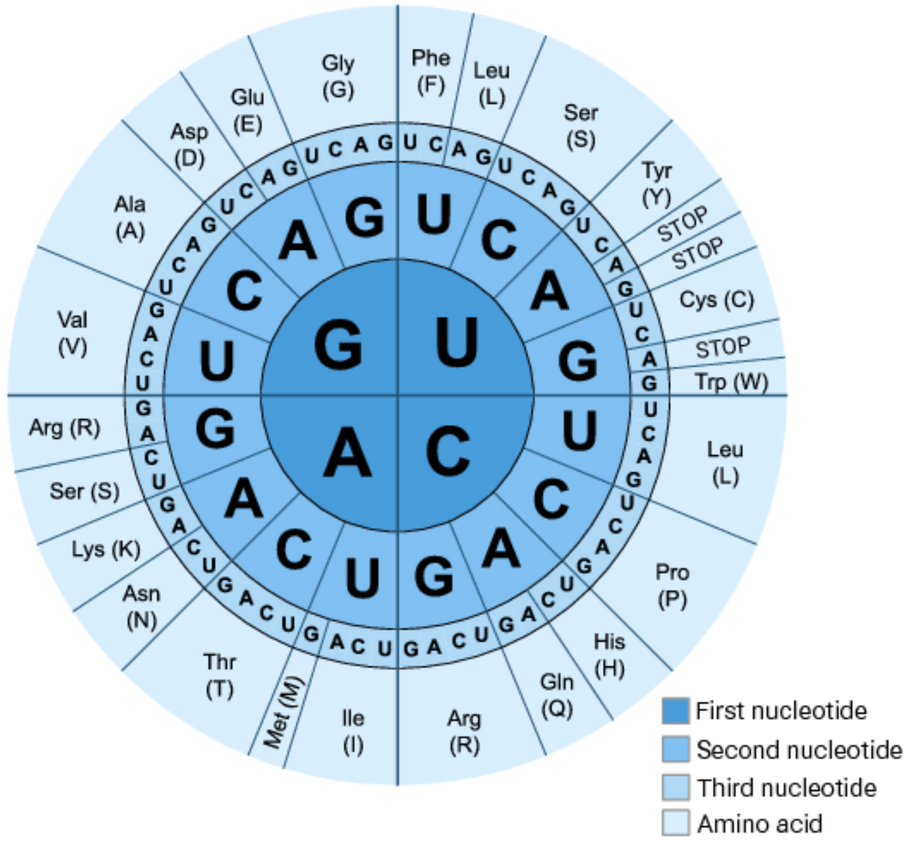
Iteration	Slice	CAAUGGCAACAUUUCAAGUCUUCCAAUAAAUAGGAC
1.	[0:3]	CAA
2.	[3:6]	UGG
3.	[6:9]	CAA

Translation of mRNA to Protein

```
mRNA = "CAAUGGCAACAUUUCAAGUCUUCCAAUAAAUAGGAC"
```

```
for index in range(0, len(mRNA), 3):
    codon = mRNA[index:index + 3]
    print(codon)
```

Translation of mRNA to Protein



Translation of mRNA to Protein

```
codon_dict = {
    "UUU": "Phe", "UUC": "Phe", "UUA": "Leu", "UUG": "Leu",
    "UCU": "Ser", "UCC": "Ser", "UCA": "Ser", "UCG": "Ser",
    "UAU": "Tyr", "UAC": "Tyr", "UAA": "STOP", "UAG": "STOP",
    "UGU": "Cys", "UGC": "Cys", "UGA": "STOP", "UGG": "Trp",
    "CUU": "Leu", "CUC": "Leu", "CUA": "Leu", "CUG": "Leu",
    "CCU": "Pro", "CCC": "Pro", "CCA": "Pro", "CCG": "Pro",
    "CAU": "His", "CAC": "His", "CAA": "Gln", "CAG": "Gln",
    "CGU": "Arg", "CGC": "Arg", "CGA": "Arg", "CGG": "Arg",
    "AUU": "Ile", "AUC": "Ile", "AUA": "Ile", "AUG": "Met",
    "ACU": "Thr", "ACC": "Thr", "ACA": "Thr", "ACG": "Thr",
    "AAU": "Asn", "AAC": "Asn", "AAA": "Lys", "AAG": "Lys",
    "AGU": "Ser", "AGC": "Ser", "AGA": "Arg", "AGG": "Arg",
    "GUU": "Val", "GUC": "Val", "GUA": "Val", "GUG": "Val",
    "GCU": "Ala", "GCC": "Ala", "GCA": "Ala", "GCG": "Ala",
    "GAU": "Asp", "GAC": "Asp", "GAA": "Glu", "GAG": "Glu",
    "GGU": "Gly", "GGC": "Gly", "GGA": "Gly", "GGG": "Gly"
}
```

Hva gjør denne koden?

```
codon_dict = {
    "UUU": "Phe", "UUC": "Phe", "UUA": "Leu", "UUG": "Leu",
    "UCU": "Ser", "UCC": "Ser", "UCA": "Ser", "UCG": "Ser",
    "UAU": "Tyr", "UAC": "Tyr", "UAA": "STOP", "UAG": "STOP",
    "UGU": "Cys", "UGC": "Cys", "UGA": "STOP", "UGG": "Trp",
    "CUU": "Leu", "CUC": "Leu", "CUA": "Leu", "CUG": "Leu",
    "CCU": "Pro", "CCC": "Pro", "CCA": "Pro", "CCG": "Pro",
    "CAU": "His", "CAC": "His", "CAA": "Gln", "CAG": "Gln",
    "CGU": "Arg", "CGC": "Arg", "CGA": "Arg", "CGG": "Arg",
    "AUU": "Ile", "AUC": "Ile", "AUA": "Ile", "AUG": "Met",
    "ACU": "Thr", "ACC": "Thr", "ACA": "Thr", "ACG": "Thr",
    "AAU": "Asn", "AAC": "Asn", "AAA": "Lys", "AAG": "Lys",
    "AGU": "Ser", "AGC": "Ser", "AGA": "Arg", "AGG": "Arg",
    "GUU": "Val", "GUC": "Val", "GUA": "Val", "GUG": "Val",
    "GCU": "Ala", "GCC": "Ala", "GCA": "Ala", "GCG": "Ala",
    "GAU": "Asp", "GAC": "Asp", "GAA": "Glu", "GAG": "Glu",
    "GGU": "Gly", "GGC": "Gly", "GGA": "Gly", "GGG": "Gly"
}

print(codon_dict["AGC"])
```

Hva gjør denne koden?

```
codon_dict = {
    "UUU": "Phe", "UUC": "Phe", "UUA": "Leu", "UUG": "Leu",
    "UCU": "Ser", "UCC": "Ser", "UCA": "Ser", "UCG": "Ser",
    "UAU": "Tyr", "UAC": "Tyr", "UAA": "STOP", "UAG": "STOP",
    "UGU": "Cys", "UGC": "Cys", "UGA": "STOP", "UGG": "Trp",
    "CUU": "Leu", "CUC": "Leu", "CUA": "Leu", "CUG": "Leu",
    "CCU": "Pro", "CCC": "Pro", "CCA": "Pro", "CCG": "Pro",
    "CAU": "His", "CAC": "His", "CAA": "Gln", "CAG": "Gln",
    "CGU": "Arg", "CGC": "Arg", "CGA": "Arg", "CGG": "Arg",
    "AUU": "Ile", "AUC": "Ile", "AUA": "Ile", "AUG": "Met",
    "ACU": "Thr", "ACC": "Thr", "ACA": "Thr", "ACG": "Thr",
    "AAU": "Asn", "AAC": "Asn", "AAA": "Lys", "AAG": "Lys",
    "AGU": "Ser", "AGC": "Ser", "AGA": "Arg", "AGG": "Arg",
    "GUU": "Val", "GUC": "Val", "GUA": "Val", "GUG": "Val",
    "GCU": "Ala", "GCC": "Ala", "GCA": "Ala", "GCG": "Ala",
    "GAU": "Asp", "GAC": "Asp", "GAA": "Glu", "GAG": "Glu",
    "GGU": "Gly", "GGC": "Gly", "GGA": "Gly", "GGG": "Gly"
}

print(codon_dict["ABC"])
```

Translation of mRNA to Protein

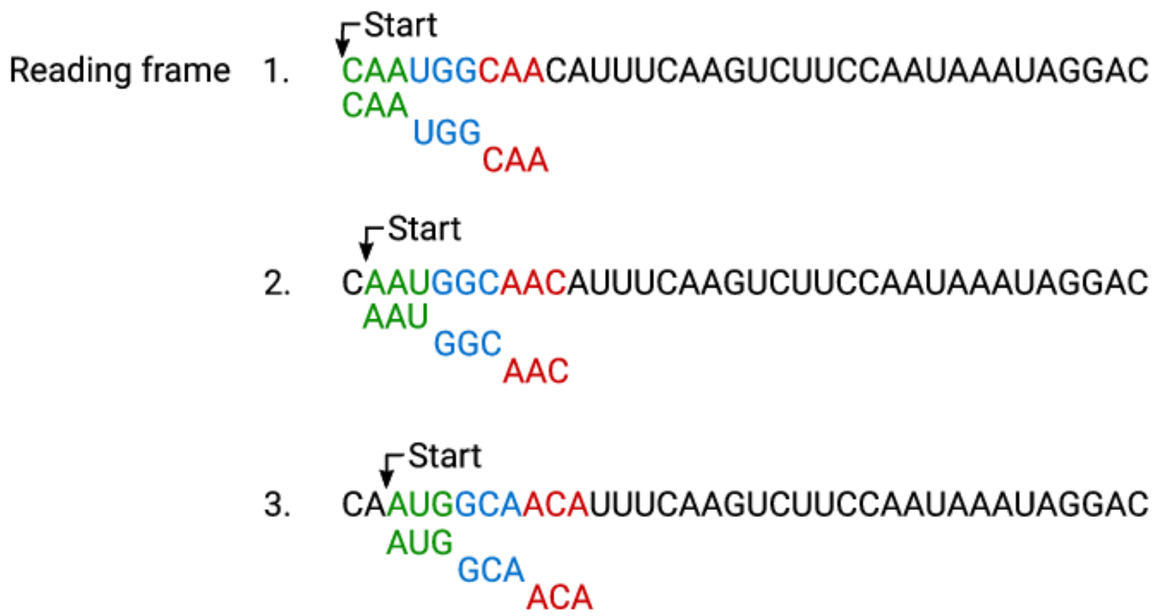
```
print(codon_dict.get("AGC"))   # exists in the dictionary
print(codon_dict.get("ABC"))   # does not exist in the dictionary
```

Translation of mRNA to Protein

```
amino_acid_sequence = []
for index in range(0, len(mRNA), 3):
    # Convert a codon to an amino acid
    # and add it to our protein
    codon = mRNA[index:index + 3]
    amino_acid = codon_dict.get(codon)
    amino_acid_sequence.append(amino_acid)

print(amino_acid_sequence)
```

```
['Gln', 'Trp', 'Gln', 'His', 'Phe', 'Lys', 'Ser', 'Ser', 'Asn', 'Lys', 'STOP',
'Asp']
```

Translation of mRNA to Protein



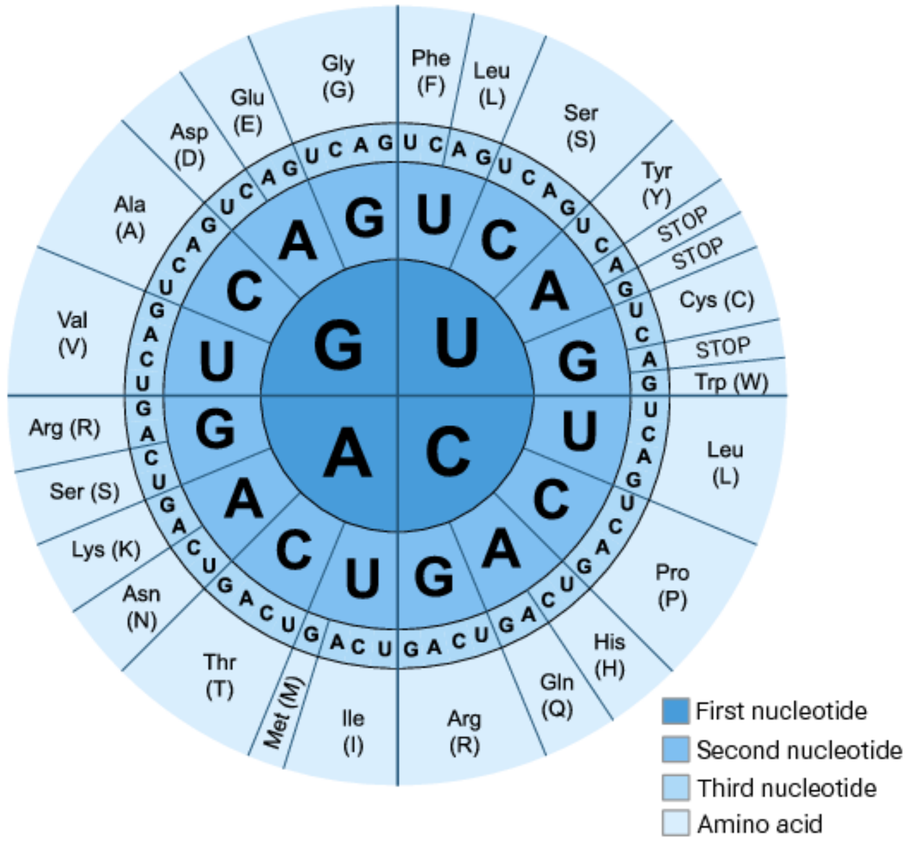
Translation of mRNA to Protein

```
mRNA = "CAAUGGCAACAUUUCAAGUCUUCCAAUAAAUAGGAC"

amino_acid_sequence = []

#                   1 instead of 0
#                   |
for index in range(1, len(mRNA), 3):
    codon = mRNA[index:index + 3]
    amino_acid = codon_dict.get(codon)
    amino_acid_sequence.append(amino_acid)

print(amino_acid_sequence)
```

```
['Asn', 'Gly', 'Asn', 'Ile', 'Ser', 'Ser', 'Leu', 'Pro', 'Ile', 'Asn', 'Arg',
None]
```

Translation of mRNA to Protein

Start and Stop codons



Translation of mRNA to Protein

Start and Stop codons

`AUG` --> 'Met' --> Start

`UAA`, `UGA` and `UAG` --> *termination codons* (Stop)

Translation of mRNA to Protein

Finding the start codon location

```
mRNA = "CAAUGGCAACAUUUCAAGUCUUCCAAUAAAUAGGAC"
```

Translation of mRNA to Protein

Finding the start codon location

```
mRNA = "CAAUGGCAACAUUUCAAGUCUUCCAAUAAAUAGGAC"

starting_offset = mRNA.index("AUG")
print(starting_offset)
```

Translation of mRNA to Protein

Translation of mRNA to Protein **algorithm**

- start at START
- translate until STOP

Translation of mRNA to Protein

Translation of mRNA to Protein **algorithm**

- start at START
- as long as the triplet is not STOP --> translate the triplet

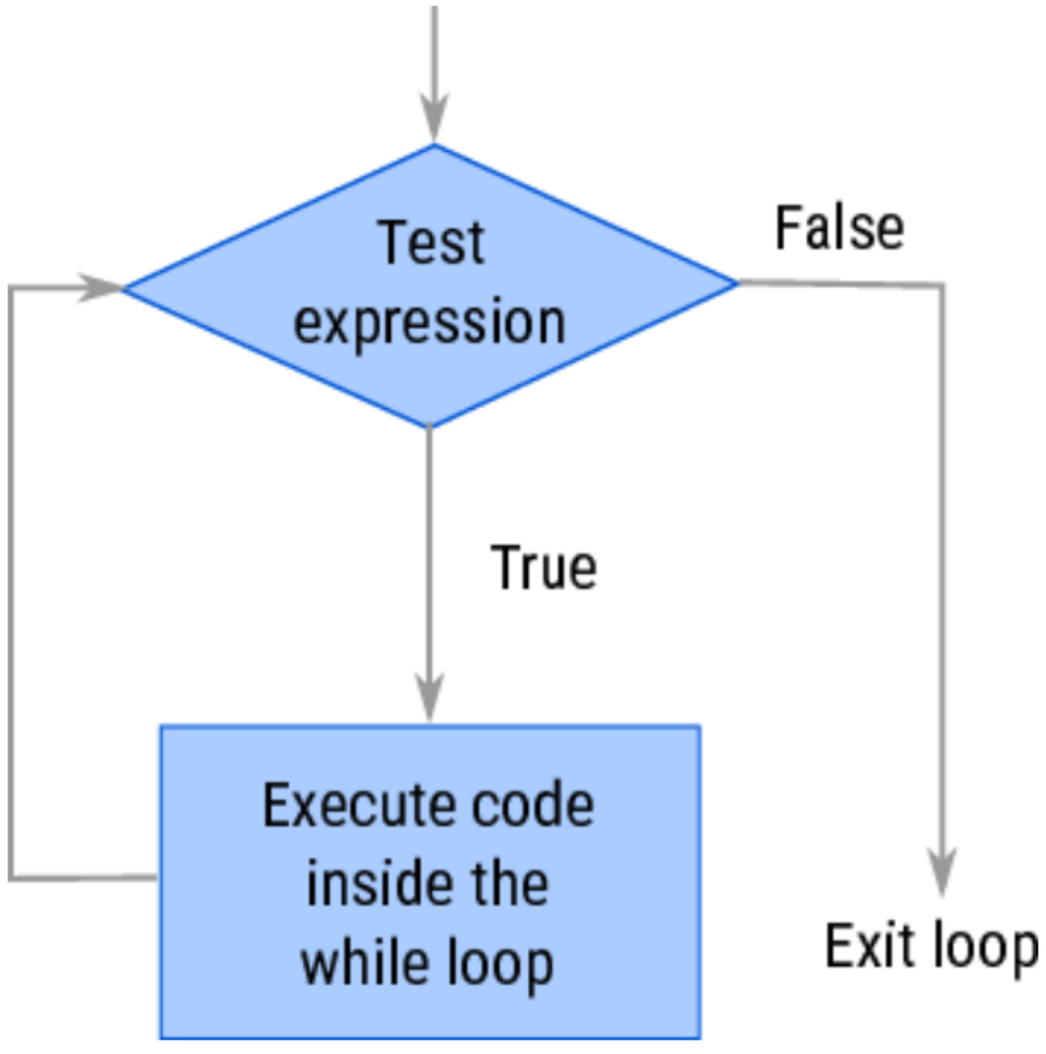
Translation of mRNA to Protein

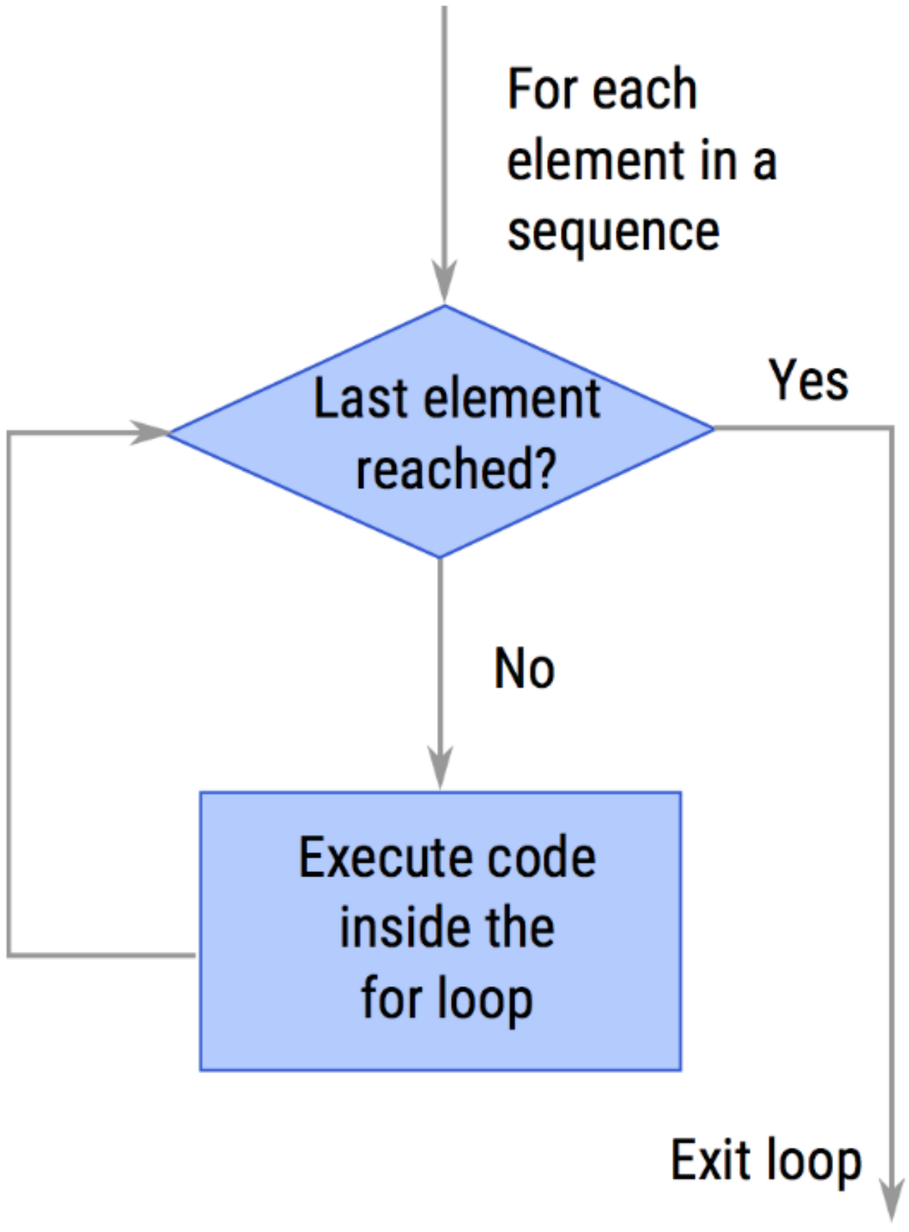
Translation of mRNA to Protein **algorithm**

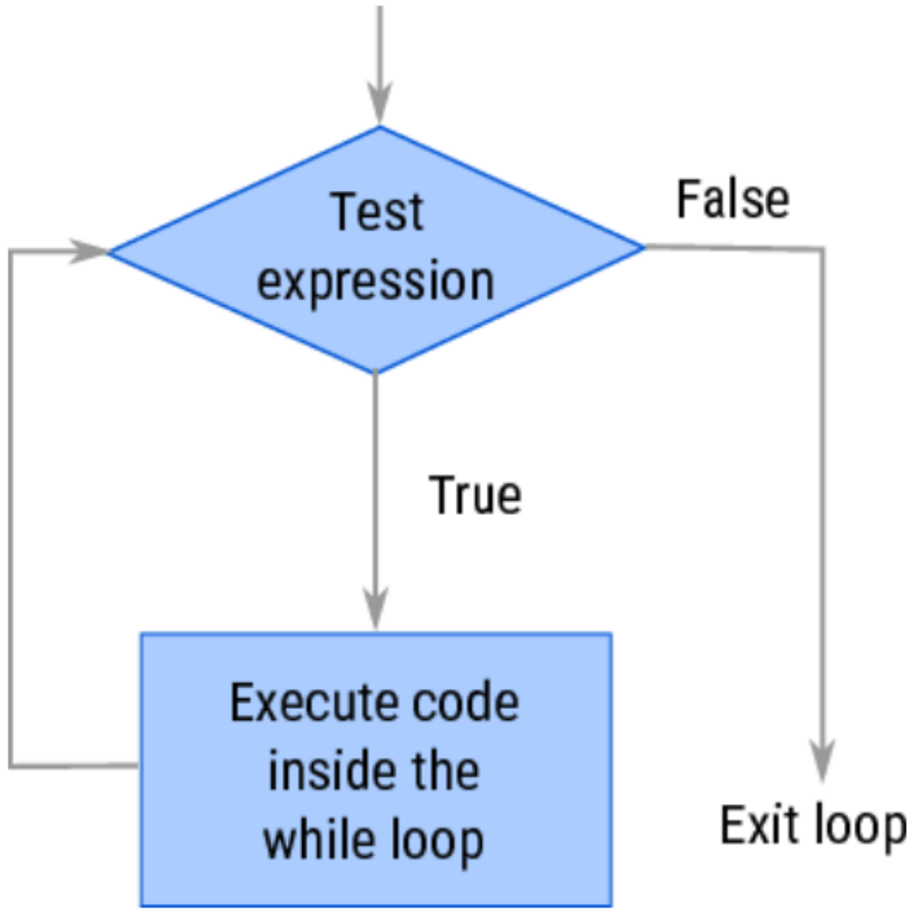
- start at START
- `while` the triplet is not STOP --> translate the triplet

`while`-loops

A `while` loop repeats a set of statements as long as a specific condition is met.




For each element in a sequence
Last element reached?
Yes
No
Execute code inside the for loop
Exit loop


Test expression
False
True
Execute code inside the while loop
Exit loop

while-loops

```
a = 0
while a < 5:
    print("Inside the loop, a is now", a)
    a = a + 1

print("After the loop, a is", a)
```

```
Inside the loop, a is now 0
Inside the loop, a is now 1
Inside the loop, a is now 2
Inside the loop, a is now 3
Inside the loop, a is now 4
After the loop, a is 5
```

Translation of mRNA to Protein

Translation of mRNA to Protein **algorithm**

- start at START
- `while` triplet is not STOP, translate each triplet

Translation of mRNA to Protein

```
mRNA = "CAAUGGCAACAUUUCAAGUCUUCCAAUAAAUAGGAC"

amino_acid = ""
while amino_acid != "STOP":
    # Convert a codon to an amino acid
    # and add it to our protein
```

Translation of mRNA to Protein

```
mRNA = "CAAUGGCAACAUUUCAAGUCUUCCAAUAAAUAGGAC"

protein = []                                          # New
amino_acid = ""
while amino_acid != "STOP":
    # Convert a codon to an amino acid
    # and add it to our protein
    codon = mRNA[index:index + 3]                     # New
    amino_acid = codon_dict.get(codon)                # New
    protein.append(amino_acid)                        # New
```

Translation of mRNA to Protein

```
mRNA = "CAAUGGCAACAUUUCAAGUCUUCCAAUAAAUAGGAC"

index = mRNA.index("AUG")                              # New
protein = []
amino_acid = ""

while amino_acid != "STOP":
    # Convert a codon to an amino acid
    # and add it to our protein
    codon = mRNA[index:index + 3]
    amino_acid = codon_dict.get(codon)
    protein.append(amino_acid)
    index += 3                                         # New

print(protein)
```

Slicing

```
my_list = [0, 2, 4, 8]
my_list[4]
```

Slicing

```
my_list = [0, 2, 4, 8]
my_list[4]
```

```
---------------------------------------------------------------------------
IndexError                                Traceback (most recent call last)
<ipython-input-75-edb684de9c31> in <module>()
      1 my_list = [0, 2, 4, 8]
----> 2 my_list[4]

IndexError: list index out of range
```

MEN

```
my_list[5:10]
```

```
[]
```

Slicing

```
mRNA = "CAAUGG"
mRNA[6]
```

```
---------------------------------------------------------------------------
IndexError                                Traceback (most recent call last)
<ipython-input-77-68c8a287a9bd> in <module>()
      1 mRNA = "CAAUGG"
----> 2 mRNA[6]

IndexError: string index out of range
```

MEN

```
mRNA[100:103]
```

```
''
```

Slicing

Slicing

- 'slicing' en liste gir altid en liste tilbake, eventuelt en tom list

Slicing

- 'slicing' en liste gir altid en liste tilbake, eventuelt en tom list
- 'slicing' en string gir altid en string tilbake, eventuelt en tom string

Translation of mRNA to Protein

The following mRNA string has no termination codon:

```
mRNA = "CAAUGGCAACAUUUCAAGUCUUCCAA"
```

Translation of mRNA to Protein

The following mRNA string has no termination codon:

```
mRNA = "CAAUGGCAACAUUUCAAGUCUUCCAA"

index = mRNA.index("AUG")                          # New
protein = []
amino_acid = ""

while amino_acid != "STOP":
    # Convert a codon to an amino acid
    # and add it to our protein
    codon = mRNA[index:index + 3]
    amino_acid = codon_dict.get(codon)
    protein.append(amino_acid)
    index += 3                                     # New

print(protein)
```

Translation of mRNA to Protein

Translation of mRNA to Protein **algorithm**

- start at START
- `while` the triplet is not STOP AND we are not at the end --> translate the triplet

Translation of mRNA to Protein

```
mRNA = "CAAUGGCAACAUUUCAAGUCUUCCAA"

index = mRNA.index("AUG")                              # New
protein = []
amino_acid = ""

while amino_acid != "STOP" and index < len(mRNA):
    # Convert a codon to an amino acid
    # and add it to our protein
    codon = mRNA[index:index + 3]
    amino_acid = codon_dict.get(codon)
    protein.append(amino_acid)
    index += 3                                         # New

print(protein)
```

Translation of mRNA to Protein

```
def translate(mRNA):
    """
    Translate a mRNA string to a protein.
    """

    codon_dict = {
        "UUU": "Phe", "UUC": "Phe", "UUA": "Leu", "UUG": "Leu",
...
...
        "GGU": "Gly", "GGC": "Gly", "GGA": "Gly", "GGG": "Gly"
    }

    index = mRNA.index("AUG")
    protein = []
    amino_acid = ""
    while amino_acid != "STOP" and index < len(mRNA):
        # Convert a codon to an amino acid
        # and add it to our protein
        codon = mRNA[index:index + 3]
        amino_acid = codon_dict.get(codon)
        protein.append(amino_acid)

        index += 3

    return protein
```

Translation of mRNA to Protein

```
mRNA = "CAAUGGCAACAUUUCAAGUCUUCCAAUAAAUAGGAC"

protein = translate(mRNA)
print(protein)
```

```
['Met', 'Ala', 'Thr', 'Phe', 'Gln', 'Val', 'Phe', 'Gln', 'STOP']
```

Utvalgte øvelser

- Exercise 4: Code tracing
- Exercise 5: Isotope decay and dating