Pristian Budi Dharmawan\_2501983105\_LA20 – BB20\_Computational Biology\_Session 2\_DNA Composition Analysis

Exercise Session 2

Google Collab Link: [Click Here](https://colab.research.google.com/drive/1pYOEngPHNn6k5Pr98QaPGksjEo3nDfIk?usp=sharing) or <https://colab.research.google.com/drive/1pYOEngPHNn6k5Pr98QaPGksjEo3nDfIk?usp=sharing>

1. **Exercise PPT Example**

Tasks

1. Which of the following sequence will have the highest GC?
2. Which of the sequence will have the higher melting temperature?
3. Which of the sequence will have the highest Molecular Weight?

Sequence

* Sequence 1 = 'ATGCATGGTGCGCGA'
* Sequence 2 = 'ATTTGTGCTCCTGGA'

Answer

Basic installation

%pip install biopython

from Bio.Seq import Seq

from Bio.SeqUtils import GC

from Bio.SeqUtils import MeltingTemp as mt

from Bio.SeqUtils import molecular\_weight

def at\_content(seq):

    result = float(seq.count('A') + seq.count('T'))/len(seq) \* 100

    return result

ex1 = Seq('ATGCATGGTGCGCGA')

ex2 = Seq('ATTTGTGCTCCTGGA')

def get\_metrics(seq):

    gc = GC(seq)

    at = at\_content(seq)

    melting = mt.Tm\_GC(seq)

    weight = molecular\_weight(seq)

    result = "GC:{}, AT:{}, Temp:{}, Weight:{}".format(gc,at,melting,weight)

    return result

Steps

Graphical user interface, application

Description automatically generated Graphical user interface, application

Description automatically generated

Text

Description automatically generated Graphical user interface, application

Description automatically generated

Graphical user interface, text

Description automatically generated

Text

Description automatically generated

1. **Exercise WGS Example**

Tasks

1. Calculate the temperature of AT and CG
2. Calculate the molecular weight of AT and CG

Sequence

* Sequence = <https://www.ncbi.nlm.nih.gov/genbank/examples.wgs/#partialcds>

Answer

Basic Installation

%pip install biopython

from Bio.Seq import Seq

from Bio.SeqUtils import GC

from Bio.SeqUtils import MeltingTemp as mt

from Bio.SeqUtils import molecular\_weight

def at\_content(seq):

    result = float(seq.count('A') + seq.count('T'))/len(seq) \* 100

    return result

WGS = Seq('TGcaaagtGGAATTCCAATTTCAACACCAGTTTTTGATGGCGCAAAAGAGCAAGATGTAACAAATATGTTAGAGCTTGCATCATTACCAAAATCTGGTCAAACAAAATTGTGGGATGGTAGAACAGGTGAAAAATTTGATAGAGAAGTCACAGTTGGCACTATTTATATGTTAAAATTACACCATCTTGTAGAAGATAAAATACACGCAAGATCTACAGGTCCTTATAGTTTAGTTACACAACAACCTCTTGGTGGTAAGGCTCAATTGGGAGGTCAACGATTTGGAGAAATGGAAGTTTGGGCTCTGGAAGCTTATGGGGCTTCTTATACTTTACAAGAAATTTTAACAGTAAAATCTGATGATGTTGCTGGTAGAGTTAAAGTTTATGAAACAATAGTAAAAGGTGAAGAGAATTTCGAGTCAGGAATACCTGAGTCATTTAATGTTTTAGTAAAAGAAATCAAAGCGCTAGCTCTTAATGTGGAGTTAAATTAAAATGAAAAAAGATATTAAAGATTTTTTTAAAGAAACTGCCATATCAGACTCTCAAAATTTTAATAGTATTAAAATTACTTTAGCAAGCCCTGAAAAGATAAAGTCATGGACTTATGGAGAAATAAAAAAACCCGAAACTATTAATTATAGAACTTTCAGACCTGAAAAAGACGGCCTATTTTGTGCGAGAATATTTGGTCCAATAAAAGATTACGAATGTTTATGTGGAAAATATAAAAGAATGAAGTTCAGAGGAATTATTTGTGAGAAGTGTGGCGTAGAGGTTACTAAATCAAATGTTCGTAGAGAAAGAATGGGGCACATCAATTTATCAACCCCAGTTGCACATATTTGGTTTTTAAAATCTTTACCAAGTAGAATTTCACTAGCTATTGATATGAAGCTTAAAGAGGTTGAAAGAGTTCTATACTTTGAAAGTTTTATTGTTATAGAGCCTGGATTAACTAGTCTTAAAAAAAATCAACTTTTAAACGAAGATGAATTAAATAAATATCAAGAGGAGTTTGGTGAAGAATCCTTTACTGCAGGAATAGGAGCAGAGGCGATACTAGAGATTTTAAAATCTATAGACTTGAATAAAGAGAGAGAAATTTTATTAAAAAATATAAATGAGACAAAATCAAAGGTTGCTGAAGAAAGATCTATAAAAAGATTAAAACTGATCGATTCATTTATTGAAACTGGTAACAAACCAGAATGGATGATTTTAACTACTATACCTGTAATACCACCAGAGTTAAGGCCACTTGTTCCTCTAGATGGAGGTAGATTTGCAACATCAGATCTAAACGATTTGTATAGAAGAGTTATAAATAGAAATAATAGATTGAAAAGATTAATGGATCTTAAAGCTCCAGATATAATTATTAGAAATGAAAAACGAATGTTGCAAGAGTCAGTGGATGCTTTATTCGATAATGGCAGAAGAGGCAGAGTAATTACAGGAACTGGTAAACGTCCATTAAAATCTTTGGCTGAAATGCTTAAAGGAaaacaaG')

def get\_metrics(seq):

    gc = GC(seq)

    at = at\_content(seq)

    melting = mt.Tm\_GC(seq)

    weight = molecular\_weight(seq)

    result = "GC:{}, AT:{}, Temp:{}, Weight:{}".format(gc,at,melting,weight)

    return result

Steps

Graphical user interface, application

Description automatically generated

Graphical user interface, text

Description automatically generated