



PennState
Huck Institutes of
the Life Sciences

University of Puerto Rico Puerto Rico - IDeA Networks of Biomedical Research Excellence



Python3 Part 1 – Crash Course in Python3 for Future STEM Coders

Judith S. Rodriguez-Martinez, M.S.

Ph.D. Candidate

Penn State University – University Park Campus
jzr5814@psu.edu



June 3, 2024 at UPR-Ciencias Médicas



- The following material is the result of PR-INBRE research and curriculum development effort to provide a set of educational materials for research training and curriculum changes in biology programs across the island to support PR-INBRE efforts to establish a Community of Practice in Bioinformatics that offers a fruitful environment to increase computational and bioinformatics skills among traditional researchers and students (undergraduate and graduate) in the island. They have been developed as a part of the NIH funded project “**Puerto Rico IDeA Network Biomedical Research Excellence (PRINBRE)**” (Award Number 5P20GM103475).
- Unless otherwise specified, all the information contained within is Copyrighted © by University of Puerto Rico. Permission is granted for use, modify, and reproduce these materials for research and teaching purposes. A copy of the modified material should be sent to help@hpcf.upr.edu.
- Most recent versions of these presentations can be found at <http://inbre.hpcf.upr.edu/>.





Competencies

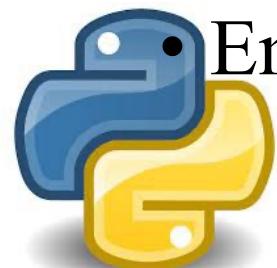
Record and write simple and common **Python scripts to deal with Bioinformatic needs for biological data analyses** using a Jupyter Notebook





Objectives

- Use **Google Colab as an environment** to practice and learn common **Python** lines of code.
- Formulate simple calculations using **Python**
- Identity the following **datatypes**: integer, float, and string



Employ **variables** for different datatypes

Objectives

- Manipulate integer, float, and string datatypes
- Use **bioinformatic tools** to find a protein-coding sequence
- Evaluate a protein-coding sequence of interest and calculate its GC content.



Target Audience

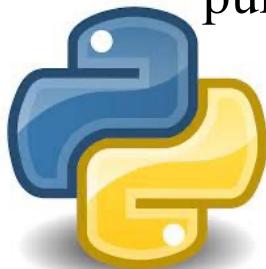
- This training is addressed to beginners, highly motivated (eager to learn what is needed in order to be competitive without the tendency to self-limit when learning computational skills by saying that is difficult) wanting to become familiar with the **Python** programming language and become the Script Master of their bioinformatic analysis.





The importance of Python

- R is great; however, you'll eventually run into the Python language
- No matter your field, python will be a necessity (chemistry, biology, engineering, ecology, mathematics, etc)
- Knowing Python gives you an edge when job searching and navigating graduate research
- The majority of bioinformatic tools are python-based and will lead you navigate under the hood code to successfully execute said program's purpose.





PennState
Huck Institutes of
the Life Sciences

Lesson 1.0.0

Connecting to Google Colab to learn common Python lines of code

(Gmail account ready for colab.research.google.com)





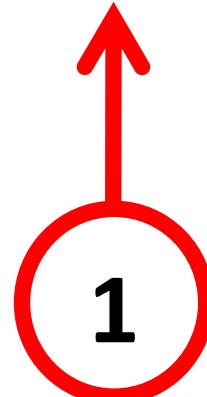
PennState
Huck Institutes of
the Life Sciences



GitHub

<https://github.com/jsrdrgz/PR->

INBRE Python Workshop



Activate the link

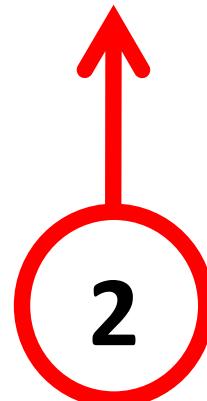


PennState
Huck Institutes of
the Life Sciences



Google Colab

<https://colab.research.google.com>



Activate the link

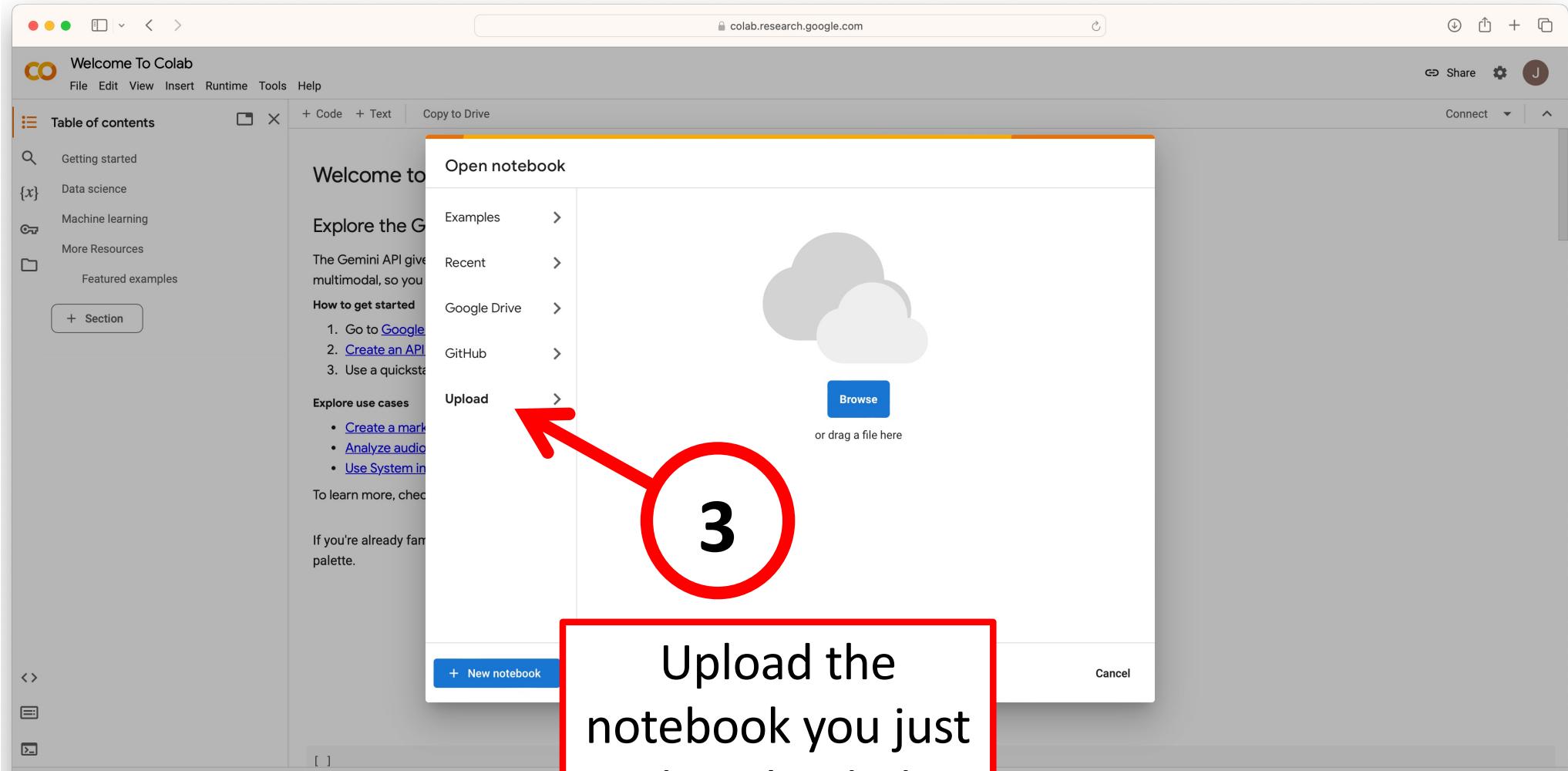


Activate colab.research.google.com

The screenshot shows the Google Colab interface. On the left, there's a sidebar with a 'Table of contents' section containing links like 'Getting started', 'Data science', 'Machine learning', 'More Resources', and 'Featured examples'. The main area displays a 'Welcome to Colab' page with sections for 'Explore the Gemini API', 'How to get started', and 'Explore use cases'. A large 'Open notebook' dialog box is overlaid on the page. This dialog has a title 'Open notebook', a central area with three cloud icons and a 'Browse' button, and a message 'or drag a file here'. At the bottom of the dialog are two buttons: '+ New notebook' on the left and 'Cancel' on the right.



Open a New notebook



Upload the
notebook you just
downloaded



Welcome to your first notebook!

The screenshot shows a Google Colab notebook titled "Student_PART_1_Python3_Workshop.ipynb". A large red circle with the number 4 is drawn around the title cell "WELCOME TO THE FIRST DAY OF OUR PYTHON CRASH COURSE!". A red arrow points from the top left towards the file tab. The notebook contains several sections and exercises:

- Table of contents**:
 - WELCOME TO THE FIRST DAY OF OUR PYTHON CRASH COURSE!
 - Lesson 1.1.0 Testing snippets of code using the Python Interpreter
 - Example 1
 - Problem 1
 - Problem 2
 - Problem 3
 - Lesson 1.2.0. Python (Integers and floats and strings...oh my!)
 - Identify datatypes of Python objects using type() functions.
 - Run this classic string!
 - Assign a variable to this string and print variable
 - Lesson 1.3.0 Evaluate the GC content of a DNA sequence
 - Create a variable for your favorite protein-coding gene sequence.
 - Get to know the CRTAM sequence
 - Find the GC content of the CRTAM sequence
 - Lesson 1.3.1 Define a function
 - Lesson 1.4.0 Running into errors
 - Proficiency Assessment





PennState
Huck Institutes of
the Life Sciences

Lesson 1.1.0

Test snippets of code
using the Python
interpreter



Python is your new **calculator**!

The image shows a screenshot of a Python Colab notebook interface. At the top, there's a navigation bar with a 'CO' logo, a file named 'Untitled0', and menu items like File, Edit, View, Help, + Code, and + Text. On the left, there are icons for search, code, and variables. A large Python logo is at the bottom left. The main area displays a list of arithmetic operators:

Addition	+
Subtraction	-
Multiplication	*
Division	/
Modulus	%
Floor division	//
Exponent	**

The entire list is enclosed in a red box.

Let's solve metric conversion problems!

The screenshot shows a Jupyter Notebook interface with the following details:

- Title Bar:** CO OX-Python3-Workshop.ipynb
- Menu Bar:** File Edit View Insert Runtime Tools Help
- Toolbar:** + Code + Text
- Table of Contents:**
 - Metric Conversion Problems
 - Example 1
- Content Area:** How many seconds(s) are in 10 milliseconds (ms)?
- Execution Area:** Contains a play button and a red-bordered "Type equation" input field.
- Red Annotations:**
 - A red circle labeled "1" with a red arrow points to the "Type equation" input field.
 - A red circle labeled "2" with a red arrow points to the "Run" button.
- Python Logo:** A blue and yellow Python logo is located at the bottom left.

Let's solve metric conversion problems!

OX-Python3-Workshop.ipynb

File Edit View Insert Runtime Tools Help All changes saved

+ Code + Text

Metric Conversion Problems

Example 1

How many seconds(s) are in 10 milliseconds (ms)?

10 * (1/0.001)

10000.0

Note: Python interprets the numbers as integers or floats

3

Answer



17

Try it on your own!

▼ Problem 1

How many $5\mu\text{L}$ are in mL?

ANSWER



Try it on your own!

▼ Problem 1

How many $5\mu\text{L}$ are in mL?

```
[ ] 5 * (0.001/1)
```

0.005



Try it on your own!

▼ Problem 2

How many kg are in 0.0034 g?

ANSWER



Try it on your own!

▼ Problem 2

How many kg are in 0.0034 g?

```
[ ] 0.0034 * (0.001/1)
```

3.4e-06



Try it on your own!

▼ Problem 3

How many moles are there in 50g of water(H₂O)?

Tip: Use variables

ANSWER



Try it on your own!

▼ Problem 3

How many moles are there in 50g of water(H₂O)?

Tip: Use variables

#Given:

mass_of_h2o = 50

molecular_mass_of_h2o = (1)*2 + 16

#Solution

```
result = mass_of_h2o/molecular_mass_of_h2o  
result
```

2.7777777777777777

Note: Facilitate
your programing
employing
variables!





PennState
Huck Institutes of
the Life Sciences

Lesson 1.2.0

Python datatypes

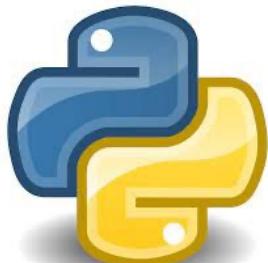


Integers and floats and strings... oh my!

Tip: Knowing the function `type(object)` of objects you are working with during coding can help you get around programming obstacles!

Datatypes

<code>int()</code>	12
<code>float()</code>	12.0
<code>str()</code>	"hola"
<code>list()</code>	[1,2,3, "hola"]
<code>dictionary()</code>	{"blue":"azul"}



Integers and floats and strings... oh my!

- ▼ Identify datatypes of Python objects using type() functions.

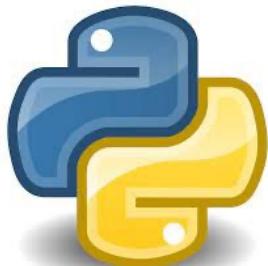
Answer



Integers and floats and strings... oh my!

- ▼ Identify datatypes of Python objects using `type()` functions.

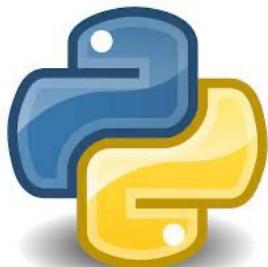
```
[15] type(mass_of_h2o)  
int
```



Integers and floats and strings... oh my!

- ▼ What is the datatype of molecular_mass_of_h2o?

Answer



Integers and floats and strings... oh my!

- ▼ What is the datatype of molecular_mass_of_h2o?

```
[✓] [16] type(molecular_mass_of_h2o)  
      int
```



Int and Float Types can be manipulated

int(*object*)
float(*object*)

How many seconds(s) are in 10 milliseconds (ms)?

✓ [23] 10 * (1/0.001)
0s

10000.0

✓ [30] seconds = 10 * (1/0.001)
int(seconds)
0s

10000

▼ Identify datatypes of Python objects using type() functions.

✓ [31] mass_of_h2o
0s

50

✓ [27] type(mass_of_h2o)
0s

int

✓ [29] float(mass_of_h2o)
0s

50.0



Integers and floats and strings... oh my!

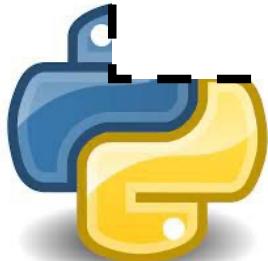
✓ [33] 'hola puerto rico'
0s

'hola puerto rico'

- ▼ Assign a variable to this string and print variable

Hint: Search for python's print() function

Answer



Integers and floats and strings... oh my!

✓ [33] 'hola puerto rico'
0s

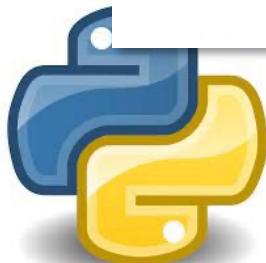
```
'hola puerto rico'
```

- ▼ Assign a variable to this string and print variable

Hint: Search for python's print() function

✓ [34] greeting = 'hello puerto rico'
0s
print(greeting)

```
hello puerto rico
```



There are a multitude of **functions** to evaluate and manipulate your **strings**

`len(str)`

`str.count("")`

`str.replace("", "")`

`str.capitalize()`

`str.split("")`

`"".join(str)`

Note: These are my favorite and most used **string functions!**

Indexing a String

Strings have a property called indexes, which are positions in the string

```
[4] sequence="ATGTGGTGG"
```



Indexing a String

Strings have a property called indexes, which are positions in the string

```
[4] sequence="ATGTGGTGG"  
sequence
```

```
' ATGTGGTGG '
```



Indexing a String

Strings have a property called indexes, which are positions in the string

```
[4] sequence="ATGTGGTGG"  
sequence
```

```
' ATGTGGTGG '  
012345678
```



Indexing a String

Strings have a property called indexes, which are positions in the string

```
[4] sequence="ATGTGGTGG"  
sequence
```

' ATGTGGTGG '
012345678
↑

The index of the
first position is 0



Indexing a String

The index of the first position is 0

'ATGTGGTGG'
012345678

[5] sequence[0]

Call the character at index 0

'A'



Indexing a String

The index of the last position is can be 8 or -1

' ATGTGGTGG '
012345678

[7] sequence[8]

Call the last character using 8

' G '

[8] sequence[-1]

Call the last character using -1

' G '

Note: Using -1 is useful when last index is unknown!

Indexing a String

Index a range of the sequence

' ATGTGGTGG '
012345678

[9] sequence[2:8]

Index is included and starts new string

Index is not included in new string

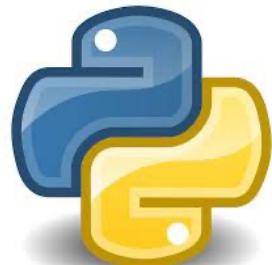




PennState
Huck Institutes of
the Life Sciences

Lesson 1.3.0

Evaluate the GC content of a DNA sequence.



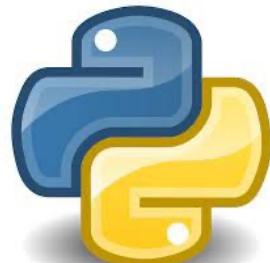
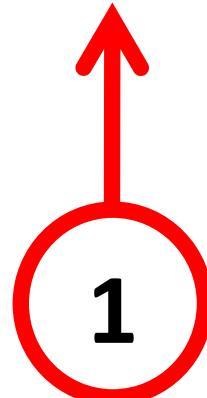


PennState
Huck Institutes of
the Life Sciences



Connect to Uniprot

<https://www.uniprot.org>



Activate uniprot.org

UniProt BLAST Align Peptide search ID mapping SPARQL Release 2023_01 | Statistics Help

Find your protein

UniProtKB Advanced | List Search Examples: Insulin, APP, Human, P05067, organism_id:9606 Feedback

UniProt is the world's leading high-quality, comprehensive and freely accessible resource of protein sequence and functional information. [Cite UniProt](#)

Proteins
UniProt Knowledgebase

Species
Proteomes

Protein Clusters
UniRef

Sequence Archive
UniParc



CRTAM is a biomarker found in prostate cancer and was a major part of my undergraduate research.

Find your protein

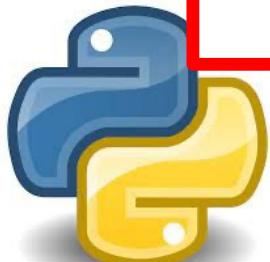
UniProtKB ▾

human crtam

Examples: Insulin, APP_Human, P05067, organism_id:9606

2

Search CRTAM



CRTAM is a biomarker found in prostate cancer and was a major part of my undergraduate research.

hiProt BLAST Align Peptide search ID mapping SPARQL UniProtKB ▾ human crtam Advanced | List Search

status
Reviewed (Swiss-Prot) (5)
Unreviewed (TrEMBL) (11)

popular organisms
uman (5)
louse (2)
ebrafish (1)

economy
ter by
oteins (1)
D structure (3)
lternative products (isoforms) (1)
lternative splicing (5)
eta strand (3)
inary interaction (5)

Q957
Q9BY
Q80U
Q8R5
A0A2R9C440
A0A2R9C440_PANPA

UniProtKB 16 results

BLAST Align Map IDs Download Add View: Cards Table Share

Select how you would like to view your results

Cards Table

3

View results

Gene Names	Organism	Length
CRTAM	Homo sapiens (Human)	395 AA
CADM1, IGSF4, GSF4A, NECL2, SYNCAM, TSLC1	Homo sapiens (Human)	442 AA
SCRIB, CRIB1, KIAA0147, LAP4, SCRIB1, VARTUL	Homo sapiens (Human)	1,111 AA
Scrib, Kiaa0147, Lap4, Scrib1	Mus musculus (Mouse)	1,111 AA
Cadm1, lgsf4, Necl2, Ra175, Syncam, SynCam1, Tslc1	Mus musculus (Mouse)	454 AA
Pan paniscus (Bushman)	Pan paniscus (Bushman)	395 AA

Cytotoxic and regulatory T cell molecule CRTAM

CRTAM is a biomarker found in prostate cancer and was a major part of my undergraduate research.

UniProt BLAST Align Peptide search ID mapping SPARQL UniProtKB ▾ human crtam Advanced | List Search    Help

Status
Reviewed (Swiss-Prot) (5)
Unreviewed (TrEMBL) (11)

BLAST Align Map IDs Download Add View: Cards Table Customize columns Share

Popular organisms
Human (5)
Mouse (2)
Zebrafish (1)

Taxonomy
Filter by taxonomy

Proteins with
3D structure (3)
Alternative products (isoforms) (5)

Feedback

UniProtKB 16 results

Entry	Entry Name	Protein Names	Gene Names	Organism	Length
O95727	CRTAM_HUMAN	Cytotoxic and regulatory T-cell molecule [...]	CRTAM	Homo sapiens (Human)	393 AA
Q14160	CADM1_HUMAN	Cell adhesion molecule 1 [...]	CADM1, IGSF4, IGSF4A, NECL2, SYNCAM, TSCLC1	Homo sapiens (Human)	442 AA
Q8K5M8	SCRIB_HUMAN	Protein scribble homolog [...]	SCRIB, CRIB1, KIAA0147, LAP4, SCRIB1, VARTUL	Homo sapiens (Human)	1,630 AA
	SCRIB_MOUSE	Protein scribble homolog [...]	Scrib, Kiaa0147, Lap4, Scrib1	Mus musculus (Mouse)	1,612 AA
	CADM1_MOUSE	Cell adhesion molecule 1 [...]	Cadm1, Igsf4, Necl2,	Mus musculus	456 AA

Activate link to Entry



CRTAM is a biomarker found in prostate cancer and was a major part of my undergraduate research.

UniProt BLAST Align Peptide search ID mapping SPARQL UniProtKB Advanced | List Search Help

|Function

095727 · CRTAM_HUMAN

Names & Taxonomy

Protein ⁱ	Cytotoxic and regulatory T-cell molecule
Gene ⁱ	CRTAM
Status ⁱ	UniProtKB reviewed (Swiss-Prot)
Organism ⁱ	Homo sapiens (Human)

Subcellular Location

Disease & Variants

PTM/Processing

Amino acids 393

Protein existenceⁱ Evidence at protein level

Annotation scoreⁱ 5/5

Expression

Interaction

Entry Feature viewer Publications External links History

Structure BLAST Align Download Add Add a publication Entry feedback

Functionⁱ

Mediates heterophilic cell-cell adhesion which regulates the activation, differentiation and tissue retention of T cells (By similarity). Interaction with CADM1 promotes natural killer (NK) cell cytotoxicity and IFNG/interferon-gamma secretion by NK cells (By similarity). In vitro as well as NK cell-mediated rejection of tumors expressing CADM1 in vivo (PubMed:15811952).

Scroll



5

CRTAM is a biomarker found in prostate cancer and was a major part of my undergraduate research.

PTM/Processing

Expression

Interaction

Structure

Family & Domains

| Sequence & Isoform

Similar Proteins

Sequence databases

CCDS | CCDS76489.1 ↗ [O95727-2]
CCDS8437.1 ↗ [O95727-1]

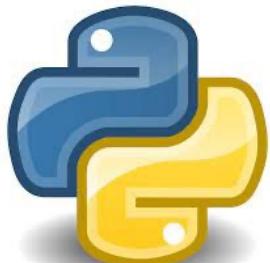
RefSeq | NP_001291711.1 ↗ NM_00
[O95727-2]
NP_062550.2 ↗ NM_01960

SEQUENCE	PROTEIN	MOLECULE TYPE
AF001622 (EMBL ↗ GenBank ↗ DDBJ ↗)	AAC80267.1 (EMBL ↗ GenBank ↗ DDBJ ↗)	mRNA
AB209830 (EMBL ↗ GenBank ↗ DDBJ ↗)	BAD93067.1 (EMBL ↗ GenBank ↗ DDBJ ↗)	mRNA
BC070266 (EMBL ↗ GenBank ↗ DDBJ ↗)	AAH70266.1 (EMBL ↗ GenBank ↗ DDBJ ↗)	mRNA

6



Activate link to EMBL



CRTAM is a biomarker found in prostate cancer and was a major part of my undergraduate research.

The screenshot shows the ENA homepage with a search bar and navigation menu. Below, a message about the Advanced Search API change is displayed. The main content area shows sequence details for AF001622.1, including organism, accession, mol type, topology, base count, dataclass, tax division, chromosome, md5 checksum, and map information. A callout box highlights the 'View' and 'Download' links for EMBL and FASTA formats, with a red arrow pointing to the FASTA link and a red circle containing the number 6. A red box at the bottom right contains the text 'Activate link to FASTA'.

ENA European Nucleotide Archive

Enter text search terms Search

Examples: histone, BN000065

AF001622 View

Examples: Taxon:9606, BN000065, PRJEB402

Home Submit ▾ Search ▾ Rulespace About ▾ Support ▾

The ENA Advanced Search API is changing on 2023-05-02! Details [here](#).

Sequence: AF001622.1

Homo sapiens class-I MHC-restricted T cell associated molecule (CRTAM) mRNA, complete cds.

Organism: Homo sapiens (human)

Accession: AF001622

Mol Type: mRNA

Topology: LINEAR;linear

Base Count: 2425

Dataclass: STD

Tax Division: HUM

Chromosome: 11

Md5 Checksum: f28e47cb1576f4d6aa50b83f6cacbd8

Map: 11q22-q23

View: EMBL FASTA

Download: EMBL FASTA

Navigation: Show

Publications: Show

Sequence Versions: View

6

Activate link to FASTA

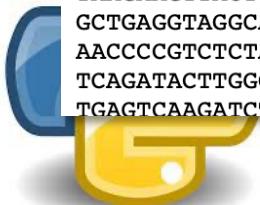
CRTAM is a biomarker found in prostate cancer and was a major part of my undergraduate research.

>ENA|AF001622|AF001622.1 Homo sapiens class-I MHC-restricted T cell associated molecule (CRTAM) mRNA, complete cds.

```
ATGTGGTGGAGACTCTCAGCTGCTGGCATGGTCCCCCTGCAAGAGGCCCTCTGACT  
AACCACACAGAAACCATCACCGTGGAGGAAGGCCAGACGCTCACTCTAAAGTGTGCACT  
TCTCTGAGGAAGAACCTCCCTCCAGTGCTGACCCCTCAGGGTTCACCATTTTTTA  
AATGAGTATCCTGCTTAAAAAATTCAAATACCAGCTTCACTCGCCAATCAG  
CTCTCCATCACTGTGCTAACGTAACCTGCAAGATGAAGCGTGTACAAGTGTACAT  
TACAGCGACTCTGTAAGCACAAGGAAGTGAAGTGATTGTGCTGGCAACTCCTTCAG  
CCAATCTGGAAGCTTCAGTTATCAGAAAGCAAAATGGAGAAGAACATGTTACTCATG  
TGCTCCACCATGAGAACGAAAGCCCCCTCCGAGATAACCTGGCTACTTGGGAATAGCATG  
GAAGTGTCCGGTGGAAAGCTCCATGAATTGAAACTGATGGGAAGAAATGTAATACTACC  
AGCACTCTCATATCCACACTTATGGCAAAAATTCAACGGTGGACTGCATTATCCGACAC  
AGAGGCCCTGCAAGGGAGAAAATAGTAGCACCCCTCCGGTTGAAGATTGTTACTGAT  
GAAGAGACAGCTCAGATGCTCTGGAGAGAAAATCTCTATCCTCAAGACCCCACAGCAG  
CCCACCACTGTCTCAGAACGGAAGATTCTAGTACATCGGAGATTGACAAGGAAGAG  
AAAGAACAAACCACTCAAGATCCTGACCTGACCACCGAACGAAATCCTCAGTATTTAGGA  
CTGGCAAGAAAGAAAAGTGGCATCCTGCTGCTCACGCTGGTGTCTTCCTCATTTCATA  
CTCTTCATCATAGTCCAGCTCTCATCATGAAGCTGAGGAAAGCACATGTGATATGGAAG  
AGAGAAAACGAAGTTCAGAACACACACTAGAAAGTTACAGATCAAGGTCAAATAATGAA  
GAAACATCATCTGAAGAGAAAATGCCAATCTCCACCCATGCGTTGATGAACATAC  
ATCACAAAGTTGACTCAGAACAAAAGAGGAAGGAAAATGTACAACATTCAA  
TTAGAAGAAAAGCACATCCAAGTACCAAGAGAGTATTGTGTAGTGCTCTGCAATGGAAC  
ATGTGATTCAGGGTGCAGTGTACCTCAGTGGACCAGCCTGGGGAGGAGCTTA  
ATTGCTGAGACATTAATAATGACCTCTTAGTGAATGCAAGATGGTGTCTCGGATAATG  
ATCTGCCCGGAGCTAGGGCAGAACATGAGGACCAAACCATGCACATAAGCTTGTAGT  
TTAAAAAAGAAAAGCAAAAAATAATTATGCCCTGACACTACTTCAGAGCAGGAGGATTCT  
ACGAAGCCTGGGGATCAGGGTCAGTGTGAGCAGCTAACATCCTACCTCAAATGGAACAG  
GATTTTTGATGCTTGTCTAATGGAACTGTTTAAAAATTTTTTCTTTTAATAT  
TTCTCTGGTCACAAAATAAGAAATTGGGATGCAAAGTACCTAAAGATCTGATCC  
TAAGAAGTTACTTCTGCCAGGGCGGGTGCATGCCGTGAACTCTAGCACTTGGGAG  
GCTGAGGTAGGCAGATCACTTGAGGTCAAGGAGTTGGAGACCAAGCCTGGCAACATAGTGA  
AACCCGCTCTACTAAAATGCAAAAATTAGCCAGGCGTAGTGGTGTGCGCACCTGTAGTC  
TCAGATACTTGGGAGGCTGAGGGTGGAGAATCGCTGAAACCTGGGAGGTGGAGATTGCA  
TGAGTCAGATCTACCAACTGAACCTCCAGCCTGGGCCAGAGGGAGACTCTGCTCAA
```

7

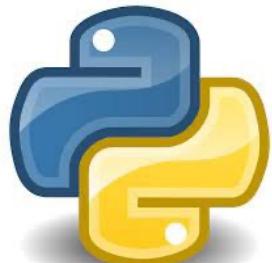
Copy Sequence



CRTAM is a biomarker found in prostate cancer and was a major part of my undergraduate research.

8

Assign sequence to variable as a string



```
sequence = """ATGTGGTGGAGAGTTCTCAGCTGCTGGCATGGTCCCTGCAAGAGGCCTCTGACT  
AACCACACAGAAACCATACCGTGGAGGAAGGCCAGACGCTCACTCTAAAGTGTGTC  
ACTCTCTGAGGAAGAACCTCCCTCCAGTGGCTGACCCCTCAGGGTCA  
CCATTTC  
AATGAGTATCCTGCTTAAAAAATTCCAATACCAGCTTCTTC  
CATCTCGGCAATCAG  
CTCTCCATCACTGTGCCTAACGTAACCTGCAAGATGAAGGCGT  
TACAAGTGCTTACAT  
TACAGCGACTCTGTAAGCACAAAGGAAGTGAAAGT  
GATTGTGCTGGCAACTCCTTC  
AAG  
CCAATCCTGGAAGCTTCAGTTATCAGAAAGCAAATGGAGAAGAACATGTTGACTCATG  
TGCTCCACCATGAGAAGCAAGCCCCCTCGCAGATAACCTGGCTACTTGGGA  
ATAGCATG  
GAAGTGTCCGGTGGACGCTCCATGA  
ATTGAAACTGATGGGAAGAAATGTA  
AACTACTACC  
AGCACTCTCATAATCCACACTTATGG  
AAAAATTCAACGGTGGACTGCATTATCCGACAC  
AGAGGCCTGCAAGGGAGAAA  
ACTAGTAGCACCCTCCGGTTGAAGATTGTTACTGAT  
TT  
TTCAGATGCTCTGGAGAGAA  
ACTCTATCCTCTCAAGACCCACAGCAG  
TGTCTCAGTAACGGAAGATTCTAGTACATCGGAGATTGACAAGGAAGAG  
CACTCAAGATCCTGACTTGACCACCGAAGCAA  
ATCCTCAGTATT  
^CCA  
GAAAAGTGGCATCCTGCTGCTACGCTGGTGC  
CTTCTCATT  
AGTCCAGCTT  
CATCATGAAGCTGAGGAAAGCACATGTGATATC  
AGTTTCAGAACACACACTAGAAAGTTACAGATCAAGGT  
CAAATAA  
TGAAGAGAAAAATGGCA  
ATCTTCCCACCC  
TATGCGTTGCATGA  
AA  
ATCACA  
AAAGTTGACTCAGAAGCA  
AAAACAAAGAGGAAGGAAA  
ATGTACAA  
CATTC  
TTAGAAGAAAAGCACATCC  
AAAGTAC  
CAGTAC  
CAGAGAGTATTGTG  
TAGTGCTCTGCA  
ATC  
ATGTGATTT  
CAGGGTTGCC  
CGAGTGT  
CACCT  
CAGTGG  
ACCAG  
GCCTGGGG  
AAGGAG  
ATTGCTGAGAC  
ATTAATGAC  
CTT  
TAGTGCA  
ATGCA  
AGATGGT  
GTC  
CTCGGA  
AT  
ATCTGCC  
CCGGAG  
CTAGGG  
CAGCA  
ACATG  
AGGG  
ACCA  
ACC  
ATGC  
ACATA  
AAGCTT  
GAGT  
TTAAAAAAAG  
AAAAG  
CAAAAAAA  
ATA  
ATT  
ATGC  
CTG  
ACACT  
ACTTCA  
GAGC  
AGGAGG  
ATTCT  
ACGAAGC  
CTTGGGG  
ATC  
AGGGT  
CAGTGT  
GAGC  
AGCTAAC  
ATC  
CTCA  
AAATGG  
AACAG
```

Note: Use triple quotations to fit the sequence in a block of code

Using the functions that you have learned, get to know the CRTAM DNA sequence.

Answer the following questions **using complete sentences**:

- What is the length of the CRTAM DNA sequence?
- How many Adenines are in your sequence?

Answer



Using the functions that you have learned, get to know the CRTAM DNA sequence.

Answer the following questions **using complete sentences**:

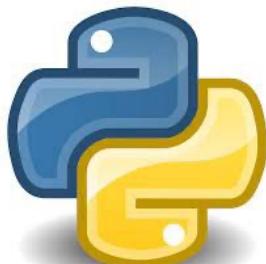
- What is the length of the CRTAM DNA sequence?
- How many Adenines are in your sequence?

```
[ ] length = len(sequence)
    print(f'The length of my DNA sequence is {length} nucleotides.')
```

The length of my DNA sequence is 2465 nucleotides.

```
[16] A_count = sequence.count('A')
    print(f'This sequence has {A_count} Adenines in total.')
```

This sequence has 772 Guanines in total.



Let's find the GC content of the CRTAM DNA sequence.

Knowledge Check

GC content

- The percentage of the proportion of guanines and cytosines to the total length of the DNA sequence.
- Provides an idea of stable a sequence is.

$$\frac{\text{Total Guanines} + \text{Total Cytosines}}{\text{Total Sequence Length}} * 100$$



Let's find the GC content of the CRTAM DNA sequence.

Knowledge Check

GC content

- The percentage of the proportion of guanines and cytosines to the total length of the DNA sequence.
- Provides an idea of stable a sequence is.

$$\frac{\text{Total Guanines} + \text{Total Cytosines}}{\text{Total Sequence Length}} * 100$$

What are we missing?



Let's find the GC content of the CRTAM DNA sequence.

Knowledge Check

GC content

- The percentage of the proportion of guanines and cytosines to the total length of the DNA sequence.
- Provides an idea of stable a sequence is.

$$\frac{\text{Total Guanines} + \text{Total Cytosines}}{\text{Total Sequence Length}} * 100$$

Return the total length of DNA sequence



Let's find the GC content of the CRTAM DNA sequence.

Knowledge Check

GC content

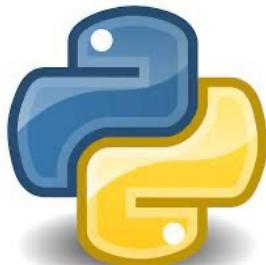
- The percentage of the proportion of guanines and cytosines to the total length of the DNA sequence.
- Provides an idea of stable a sequence is.

$$\frac{\text{Total Guanines} + \text{Total Cytosines}}{\text{Total Sequence Length}} * 100$$

```
[10] length = len(sequence)
     print(f'The length of my DNA sequence is {length} nucleotides.')
```

The length of my DNA sequence is 2465 nucleotides.

Tip: Using string formatting eases mixing different datatypes together



Let's find the GC content of the CRTAM DNA sequence.

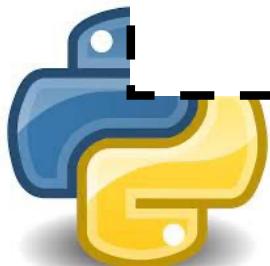
Knowledge Check

GC content

- The percentage of the proportion of guanines and cytosines to the total length of the DNA sequence.
- Provides an idea of stable a sequence is.

$$\frac{\text{Total Guanines} + \text{Total Cytosines}}{\text{Total Sequence Length}} * 100$$

Return total Guanines and Cytosines of the DNA sequence



Let's find the GC content of the CRTAM DNA sequence.

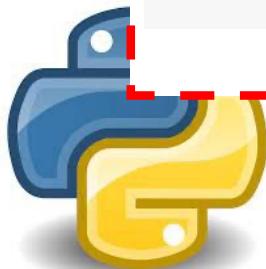
Knowledge Check

GC content

- The percentage of the proportion of guanines and cytosines to the total length of the DNA sequence.
- Provides an idea of stable a sequence is.

$$\frac{\text{Total Guanines} + \text{Total Cytosines}}{\text{Total Sequence Length}} * 100$$

```
[14] G_count = sequence.count('G')
     C_count = sequence.count('C')
     print(f'This sequence has {G_count} Guanines in total.')
     print(f'This sequence has {C_count} Cytosines in total.'
```



Let's find the GC content of the CRTAM DNA sequence.

Knowledge Check

GC content

- The percentage of the proportion of guanines and cytosines to the total length of the DNA sequence.
- Provides an idea of stable a sequence is.

$$\frac{\text{Total Guanines} + \text{Total Cytosines}}{\text{Total Sequence Length}} * 100$$

Return GC Content



Let's find the GC content of the CRTAM DNA sequence.

Knowledge Check

GC content

- The percentage of the proportion of guanines and cytosines to the total length of the DNA sequence.
- Provides an idea of stable a sequence is.

$$\frac{\text{Total Guanines} + \text{Total Cytosines}}{\text{Total Sequence Length}} * 100$$

```
[12] GC_content = (G_count + C_count)/length*100  
      print(f'The GC content of my sequence is {GC_content}')
```

```
The GC content of my sequence is 42.799188640973625
```



Let's find the GC content of the CRTAM DNA sequence.

Overall Code

```
✓ [13] length = len(sequence)
0s   print(f'The length of my DNA sequence is {length} nucleotides.')
```

The length of my DNA sequence is 2465 nucleotides.

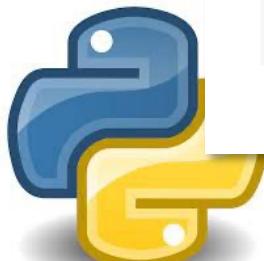
```
✓ [14] G_count = sequence.count('G')
0s   C_count = sequence.count('C')
      print(f'This sequence has {G_count} Guanines in total.')
      print(f'This sequence has {C_count} Cytosines in total.')
```

This sequence has 537 Guanines in total.

This sequence has 518 Cytosines in total.

```
✓ [15] GC_content = (G_count + C_count)/length*100
0s   print(f'The GC content of my sequence is {GC_content}')
```

The GC content of my sequence is 42.799188640973625





PennState
Huck Institutes of
the Life Sciences

Lesson 1.3.1

Define a function that returns the GC content of a DNA sequence.



Define a function that calculates GC content

```
[21] def find_GC_content(a_sequence):
    length = len(sequence)
    G_count = sequence.count('G')
    C_count = sequence.count('C')
    GC_content = (G_count + C_count)/length*100
    return(GC_content)
```



Define a function that calculates GC content

```
[21] def find_GC_content(a_sequence):
```



In the first line, a function is initiated by using **def**



Define a function that calculates GC content

```
[21] def find_GC_content(a_sequence):
```



Name the function
as you see fit



Define a function that calculates GC content

```
[21] def find_GC_content(a_sequence)
```



Identify if the function takes **input**. Here, our function just needs one **input**. The function needed as **input** is a DNA sequence in order to return the GC content.

DON'T FORGET PARENTHESES!



Define a function that calculates GC content

```
[21] def find_GC_content(a_sequence):
```

A red square box surrounds the colon character (:) at the end of the function definition line.

End it with a **colon**

A red arrow points upwards from a red rectangular box containing the text "End it with a colon" towards the red square box around the colon character.

Define a function that calculates GC content

```
[21] def find_GC_content(a_sequence):  
    length = len(sequence)  
    G_count = sequence.count('G')  
    C_count = sequence.count('C')  
    GC_content = (G_count + C_count)/length*100
```



**Body of the function
defined **dictates**
what the function is
doing**



Define a function that calculates GC content

```
[21] def find_GC_content(a_sequence):  
    length = len(sequence)  
    G_count = sequence.count('G')  
    C_count = sequence.count('C')  
    GC_content = (G_count + C_count)/length*100  
    return(GC_content)
```



Using the **return** function
will give you the result of the
function when you run it

**DON'T FORGET
PARENTHESES!**



Define a function that calculates GC content

```
[21] def find_GC_content(a_sequence):
    length = len(sequence)
    G_count = sequence.count('G')
    C_count = sequence.count('C')
    GC_content = (G_count + C_count)/length*100
    return(GC_content)
```

```
[22] find_GC_content(sequence)
```

42.799188640973625

Tip: Functions are a great way to control code when running a repeated block of code more than once on different datatypes!





PennState
Huck Institutes of
the Life Sciences

Lesson 1.4.0

Running into errors





Uh oh! Ran into an error!

```
[44] length = len(sequence)
    G_count = sequence.count('G')
    C_count = sequence.count('C')
    GC_content = (G_count + C_count)/length

    print("My sequence is "+str(length)+" nucleotides long.")
    print("There are "+str(G_count)+" guanines and "+C_count+" cytocines.")
    print("The GC content of my sequence is "+str(GC_content)+".")
```

My sequence is 2508 nucleotides long.

TypeError

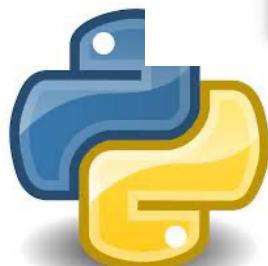
Traceback (most recent call last)

[<ipython-input-44-b05e980949a1>](#) in <module>

```
      5
      6 print("My sequence is "+str(length)+" nucleotides long.")
----> 7 print("There are "+str(G_count)+" guanines and "+C_count+" cytocines.")
      8 print("The GC content of my sequence is "+str(GC_content)+".")
```

TypeError: can only concatenate str (not "int") to str

SEARCH STACK OVERFLOW



Where is my **error**?



Debug and rerun program



PennState
Huck Institutes of
the Life Sciences

```
[44] length = len(sequence)
    G_count = sequence.count('G')
    C_count = sequence.count('C')
    GC_content = (G_count + C_count)/length

    print("My sequence is "+str(length)+" nucleotides long.")
    print("There are "+str(G_count)+" guanines and "+C_count+" cytocines.")
    print("The GC content of my sequence is "+str(GC_content)+".")
```

My sequence is 2508 nucleotides long.

```
-----  
TypeError                                 Traceback (most recent call last)  
<ipython-input-44-b05e980949a1> in <module>  
      5  
      6     print("My sequence is "+str(length)+" nucleotides long.")  
----> 7     print("There are "+str(G_count)+" guanines and "+C_count+" cytocines.")  
      8     print("The GC content of my sequence is "+str(GC_content)+".")  
  
TypeError: can only concatenate str (not "int") to str
```

SEARCH STACK OVERFLOW



Where is my error?



Error Corrected



PennState
Huck Institutes of
the Life Sciences

```
[45] length = len(sequence)
G_count = sequence.count('G')
C_count = sequence.count('C')
GC_content = (G_count + C_count)/length

print("My sequence is "+str(length)+" nucleotides long.")
print("There are "+str(G_count)+" guanines and "+str(C_count)+" cytocines.")
print("The GC content of my sequence is "+str(GC_content)+".")
```

My sequence is 2508 nucleotides long.
There are 549 guanines and 528 cytocines.
The GC content of my sequence is 0.42942583732057416.

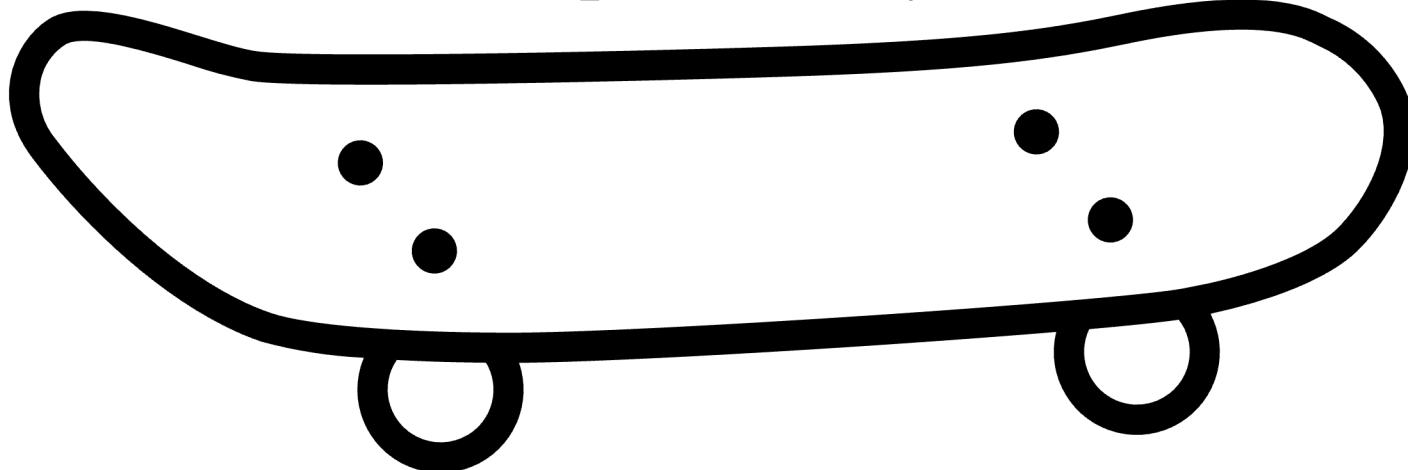




PennState
Huck Institutes of
the Life Sciences

Proficiency Assessment

Show off your new skills using Google COLAB with a proficiency assessment



Develop a block of code

- 1. Find a DNA sequence of your interest using Uniprot and cross referencing a database. Assign your sequence a variable.**

- 2. Define and return the following functions:**
 - 1. Total number of Guanines**
 - 2. Total number of Cytosines**
 - 3. Total number of Adenines**
 - 4. Total number of Uridines**
 - 5. A+T/G+C ratio**

