

# HUMBIO51- Class 2

What does it mean for a gene to be expressed?

How can we write out the sequence of the mRNA transcript for a DNA sequence?

Annette Salmeen & Anshul Kundaje

TA: Anna Shcherbina

# Contact Information & Office Hours

Annette Salmeen

e-mail: [asalmeen@stanford.edu](mailto:asalmeen@stanford.edu)

Office: Main Quad Bldg 20-21F

Office Hours:

Tu 4:00-5:00PM

Fri 10:00-11:00AM

Or by appointment

Anshul Kundaje

e-mail: [akundaje@stanford.edu](mailto:akundaje@stanford.edu)

Office: Lane L301B

Office Hours:

By appointment

Anna Shcherbina

e-mail:

[annashch@stanford.edu](mailto:annashch@stanford.edu)

Office hours:

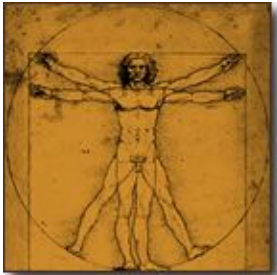
Wednesday 10:00 - 11:00 am

# Agenda

- Course introduction & Objectives
- Complementary sequences
- String slicing
- Writing out sequences to a file
- Writing out the RNA transcription product of a DNA sequence
- RNA splicing
- Finding start and stop codons in a mRNA sequence

# Unit 1:

## What are genes, DNA, RNA and proteins? Getting Started with Python



Human Genome  
Project

2003

2012

2015

2007

# Two options for accessing Jupyter Notebooks

- Via the class server on the Google Cloud Platform (link posted on Canvas)

<https://console.cloud.google.com/home/dashboard?project=gbsc-gcp-class-humbio51-aut17&organizationId=302681460499>

- By installing Anaconda and running Jupyter Notebooks from your computer.

<https://www.anaconda.com/download/>

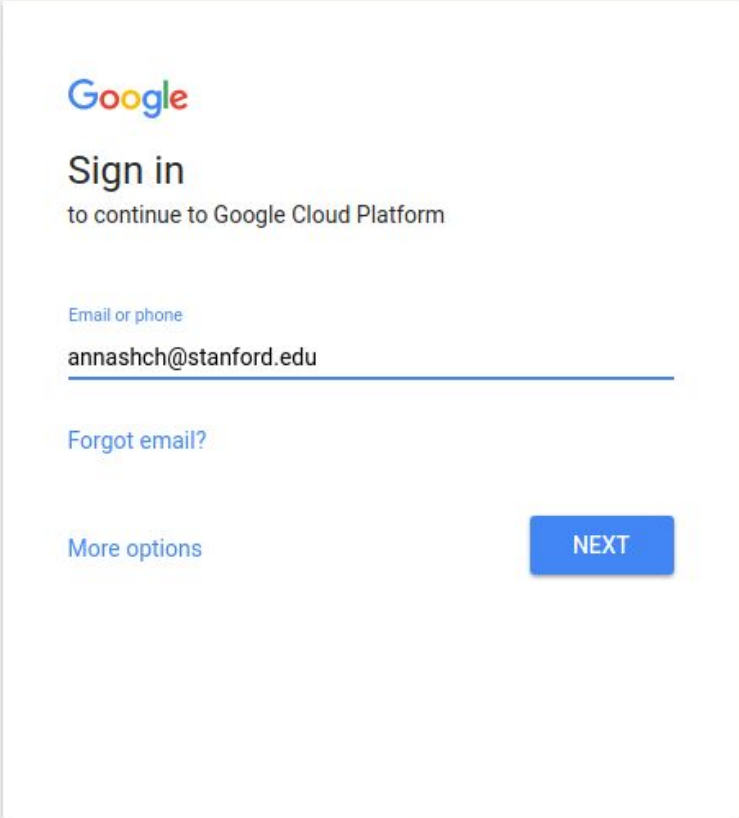
Note: this will also require you to install Python libraries and other programs

# How to use the Google Cloud Platform:

## Setting up your account

1. <https://console.cloud.google.com/start>

2. Specify your stanford.edu e-mail account as the login account

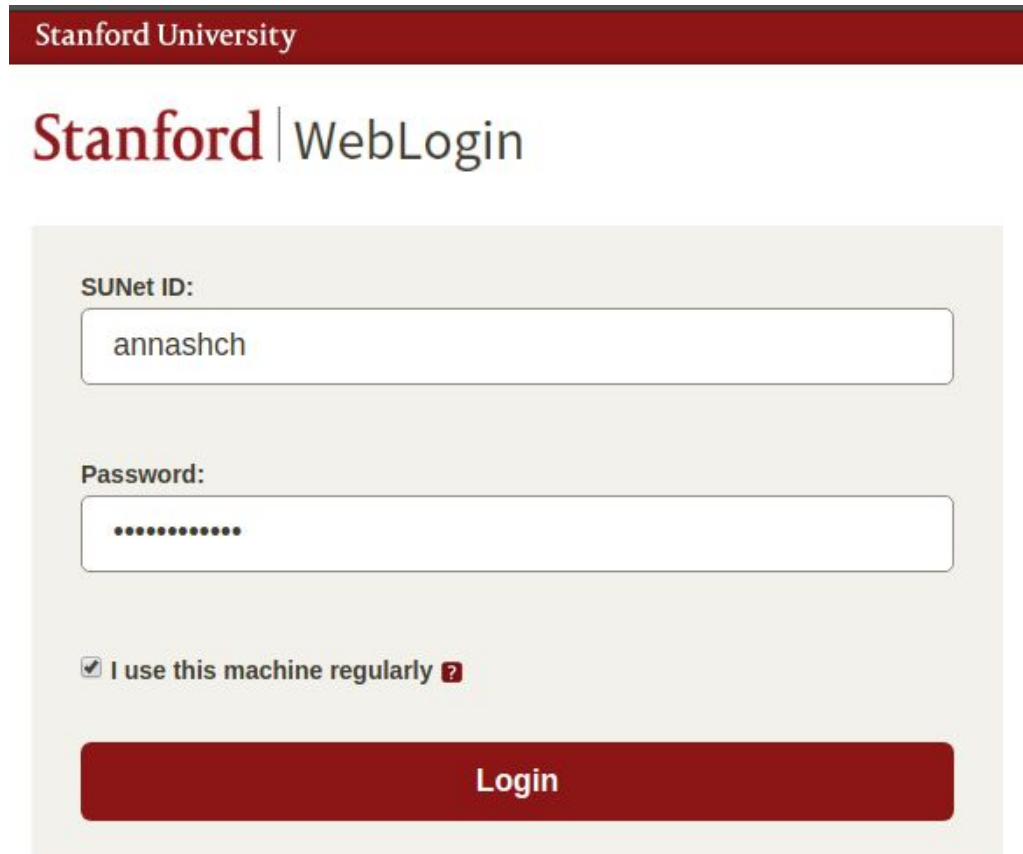


The screenshot shows the Google Cloud Platform sign-in interface. At the top is the Google logo, followed by the text "Sign in to continue to Google Cloud Platform". Below this is a label "Email or phone" and a text input field containing "annashch@stanford.edu". To the left of the input field are two links: "Forgot email?" and "More options". To the right of the input field is a blue "NEXT" button. At the bottom of the page, there is a footer with "English (United States)" and a dropdown arrow, and links for "Help", "Privacy", and "Terms".

# How to use the Google Cloud Platform:

## Setting up your account

3. Complete the Stanford WebLogin prompt



The image shows the Stanford University WebLogin interface. At the top is a dark red header with the text "Stanford University" in white. Below this is the "Stanford | WebLogin" logo, with "Stanford" in red and "WebLogin" in grey. The main login area is a light beige box containing two input fields: "SUNet ID:" with the text "annashch" and "Password:" with masked characters. Below these is a checkbox labeled "I use this machine regularly" with a red question mark icon. At the bottom is a large red "Login" button.

Stanford University

Stanford | WebLogin

SUNet ID:

annashch

Password:

.....

☒ I use this machine regularly ?

Login

# How to use the Google Cloud Platform:

## Setting up your account

4. You will see the GCP portal

The screenshot shows the Google Cloud Platform (GCP) 'Getting started' page. The header is a blue bar with the 'Google Cloud Platform' logo, a 'Select a project' dropdown, a search bar, and user account icons. Below the header, the page is titled 'Getting started'. The main content area features several interactive tiles:

- Try Compute Engine**: A blue tile with a 'Get started' button. Description: 'Spin up virtual machines using Google Compute Engine, Node.js, and MongoDB to create a To-Do app in this guided walkthrough.'
- Learn to use Cloud Storage**: A white tile with a 'Get started' button. Description: 'Cloud Storage is a powerful and simple storage service. In this tutorial you'll learn the basics by creating a storage bucket, and then uploading and sharing a sample file as a public URL link.'
- Learn Google Cloud Platform**: A white tile with a 'Get Started' button. Description: 'Take an interactive tutorial now and learn how to deploy and build simple applications.'
- Use Google APIs**: A white tile with an 'API Enable and manage APIs' button. Description: 'Enable APIs, create credentials, and track your usage.'
- Create a Cloud SQL instance**: A white tile with a 'Get started' button. Description: 'Cloud SQL is a MySQL database that runs in Google's cloud, with no installation or maintenance required.'
- Try App Engine**: A white tile with a 'Get started' button and a dropdown arrow. Description: 'Create and deploy a Hello World app.'
- Documentation**: A section with links to 'Learn about Compute Engine', 'Learn about Cloud Storage', and 'Learn about App Engine', each with an external link icon.
- Create an empty project**: A white tile with a plus icon and a button.

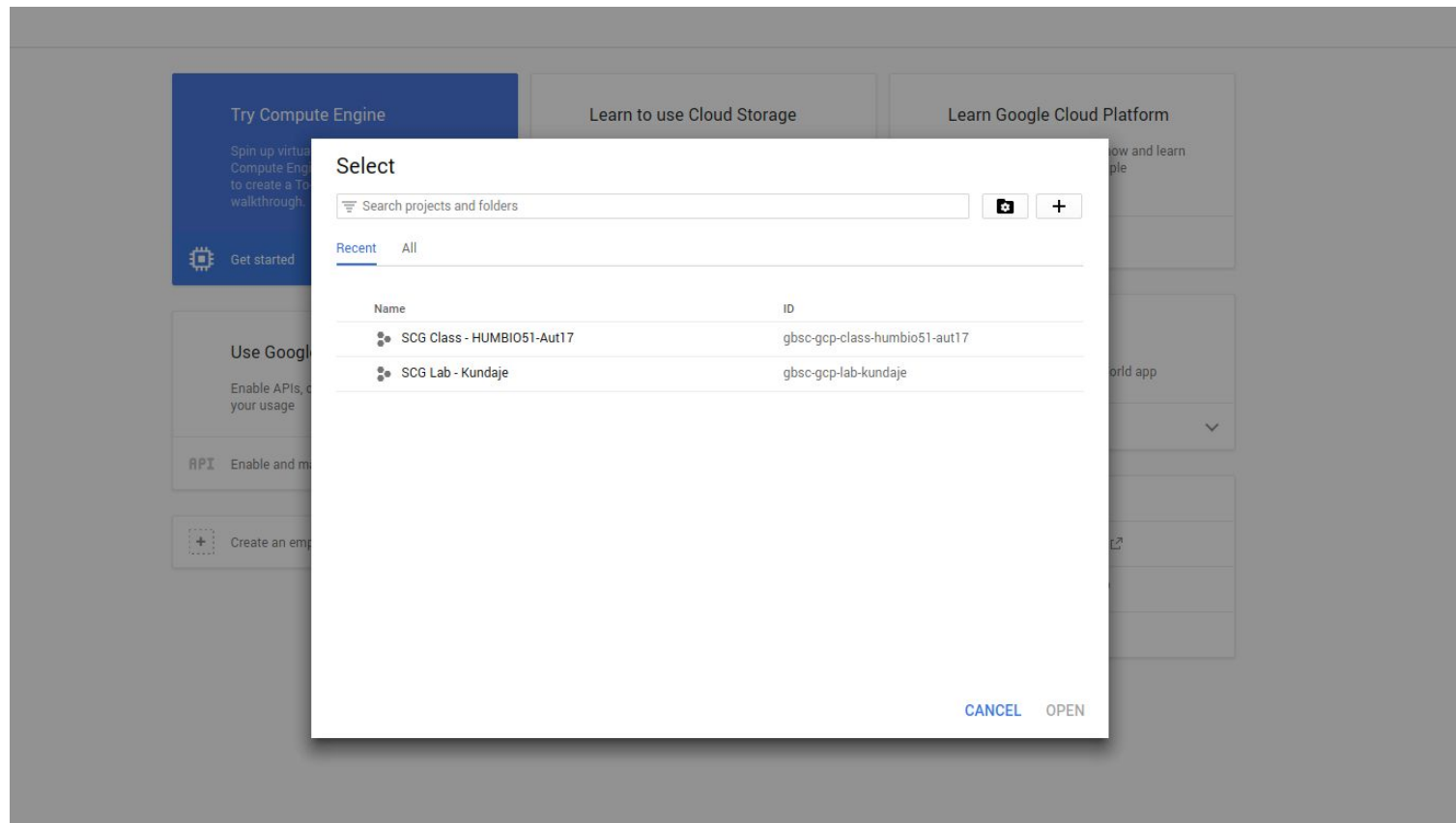
At the bottom center, there is a 'VIEW MORE' link.



# How to use the Google Cloud Platform:

## Setting up your account

4. In the top blue bar, where it says “Select a project”, select “SCG Class-HUMBIO51-Aut17”



# How to use the Google Cloud Platform:

## Setting up your account

You have now successfully set up your GCP account. Next time, to access your account, simply go to <https://console.cloud.google.com> and follow the steps on the next slides.

# How to use the Google Cloud Platform

The screenshot shows the Google Cloud Platform dashboard for the project 'SCG Class - HUMBIO51-A...'. The left-hand navigation menu is expanded, showing the 'Compute Engine' section. A red '1)' is placed next to the 'Compute Engine' link. A dropdown menu is open for 'Compute Engine', showing various options. A red '2)' is placed next to the 'VM instances' option in the dropdown. The main dashboard area displays 'Project info', 'Compute Engine' status (CPU usage graph), 'Google Cloud Platform status', 'Billing' (estimated charges), 'Error Reporting', and 'News'.

Google Cloud Platform SCG Class - HUMBIO51-A... [CUSTOMIZE](#)

**Home** DASHBOARD ACTIVITY

Cloud Launcher  
Billing  
APIs & services  
Support  
Getting started  
IAM & admin

COMPUTE

1) **Compute Engine**

- 2) VM instances
- Instance groups
- Instance templates
- Disks
- Snapshots
- Images
- Committed use discounts
- Metadata
- Health checks
- Zones
- Operations
- Quotas
- Settings

Project info  
Project name: SCG Class - HUMBIO51-Aut17  
Project ID: gbsc-gcp-class-humbio51-aut17  
Project number: 499612801043

Compute Engine  
CPU (%)  
0.1  
0.08  
0.06  
0.04  
0.02  
Sep 23, 3:30 PM Sep 23, 4:27 PM  
CPU: 0.075  
Go to the Compute Engine dashboard

Google Cloud Platform status  
All services normal  
Go to Cloud status dashboard

Billing  
Estimated charges: \$0.00  
For the billing period Sep 1 – 23, 2017  
View detailed charges

Error Reporting  
No sign of any errors. Have you set up Error Reporting?  
Learn how to set up Error Reporting

News  
Committed use discounts for Google Compute Engine now generally available

- 1) Click where it says Compute Engine
- 2) Click VM instances

# Turn your instance on if needed

VM instances [+ CREATE INSTANCE](#) [IMPORT VM](#) [REFRESH](#) [START](#) [STOP](#) [RESET](#) [DELETE](#)

4)

Filter VM instances Columns

<input type="checkbox"/> Name ^	Zone	Recommendation	Internal IP	External IP	Network	Labels	Connect
<input checked="" type="checkbox"/> <input type="radio"/> instance-1	us-west1-a		10.138.0.14	None	default		SSH <span>⌵</span> <span>⋮</span>

3)

3) check the box next to your instance

4) click on "START"

# How to use the Google Cloud Platform

Google Cloud Platform SCG Class - HUMBI051-A...

Compute Engine

VM instances

CREATE INSTANCE IMPORT VM REFRESH START STOP RESET DELETE HIDE INFO PANEL

Filter VM instances

Name	Zone	Recommendation	Internal IP	External IP	Connect
instance-master	us-west1-a		10.138.0.2	35.197.55.121	SSH

Select an instance

LABELS MONITORING

Labels help organize your resources (e.g., cost\_center:sales or env:prod).

No instances selected.

5) Click on the external IP address.

# How to use the Google Cloud Platform

6) The password for your GCP instance is **humbio**  
or whatever you changed it to.

**NOTE YOU SHOULD CHANGE YOUR PASSWORD!!**

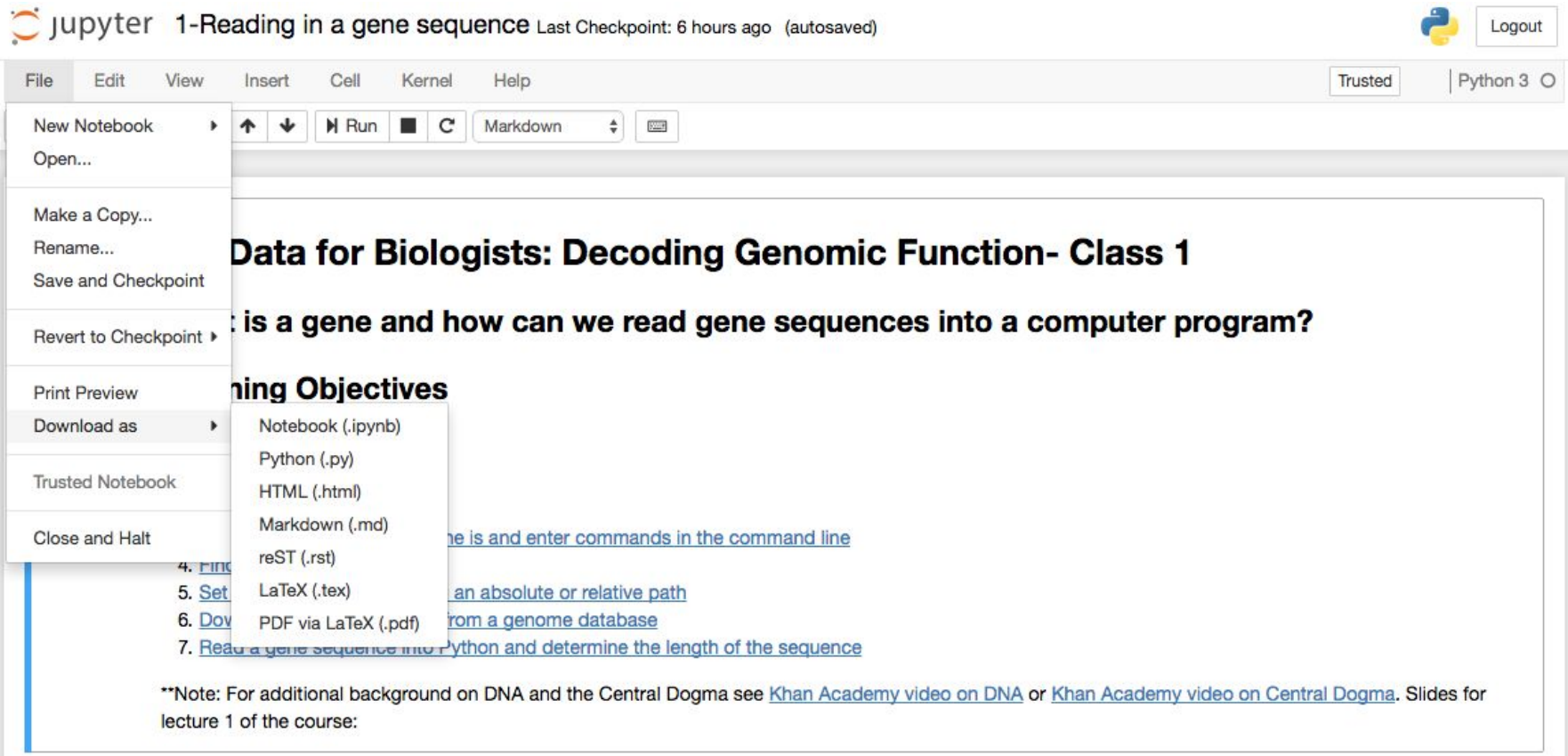


**6)**

Password:

Log in

# How to save an assignment from the Google Cloud Platform



The screenshot shows a Jupyter Notebook interface. At the top, the title bar reads "jupyter 1-Reading in a gene sequence Last Checkpoint: 6 hours ago (autosaved)". On the right, there is a "Logout" button. Below the title bar is a menu bar with "File", "Edit", "View", "Insert", "Cell", "Kernel", and "Help". To the right of the menu bar are "Trusted" and "Python 3" buttons. The "File" menu is open, showing options: "New Notebook", "Open...", "Make a Copy...", "Rename...", "Save and Checkpoint", "Revert to Checkpoint", "Print Preview", "Download as", "Trusted Notebook", and "Close and Halt". The "Download as" option is selected, and a sub-menu is open showing file formats: "Notebook (.ipynb)", "Python (.py)", "HTML (.html)", "Markdown (.md)", "reST (.rst)", "LaTeX (.tex)", and "PDF via LaTeX (.pdf)". The background content of the notebook is titled "Data for Biologists: Decoding Genomic Function- Class 1" and includes a question "What is a gene and how can we read gene sequences into a computer program?" and a section "Learning Objectives" with a list of tasks. A note at the bottom states: "\*\*Note: For additional background on DNA and the Central Dogma see [Khan Academy video on DNA](\"#\") or [Khan Academy video on Central Dogma](\"#\"). Slides for lecture 1 of the course:

Assignments need to be submitted on Canvas!

# Don't forget to turn your instance off when you are not using it!

Google Cloud Platform SCG Class - HUMBI051-A...

Compute Engine VM instances

CREATE INSTANCE IMPORT VM REFRESH START STOP RESET DELETE HIDE INFO PANEL

VM instances

Filter VM instances Columns

Name	Zone	Recommendation	Internal IP	External IP	Connect
<input type="checkbox"/> instance-master	us-west1-a		10.138.0.2	35.197.55.121	SSH

Select an instance

LABELS MONITORING

Labels help organize your resources (e.g., cost\_center:sales or env:prod).

No instances selected.

6) Click on “STOP”.

If your instance has been running for more than 12 hours, you will receive a reminder from Google to turn it off.



# How to save an assignment from the Google Cloud Platform

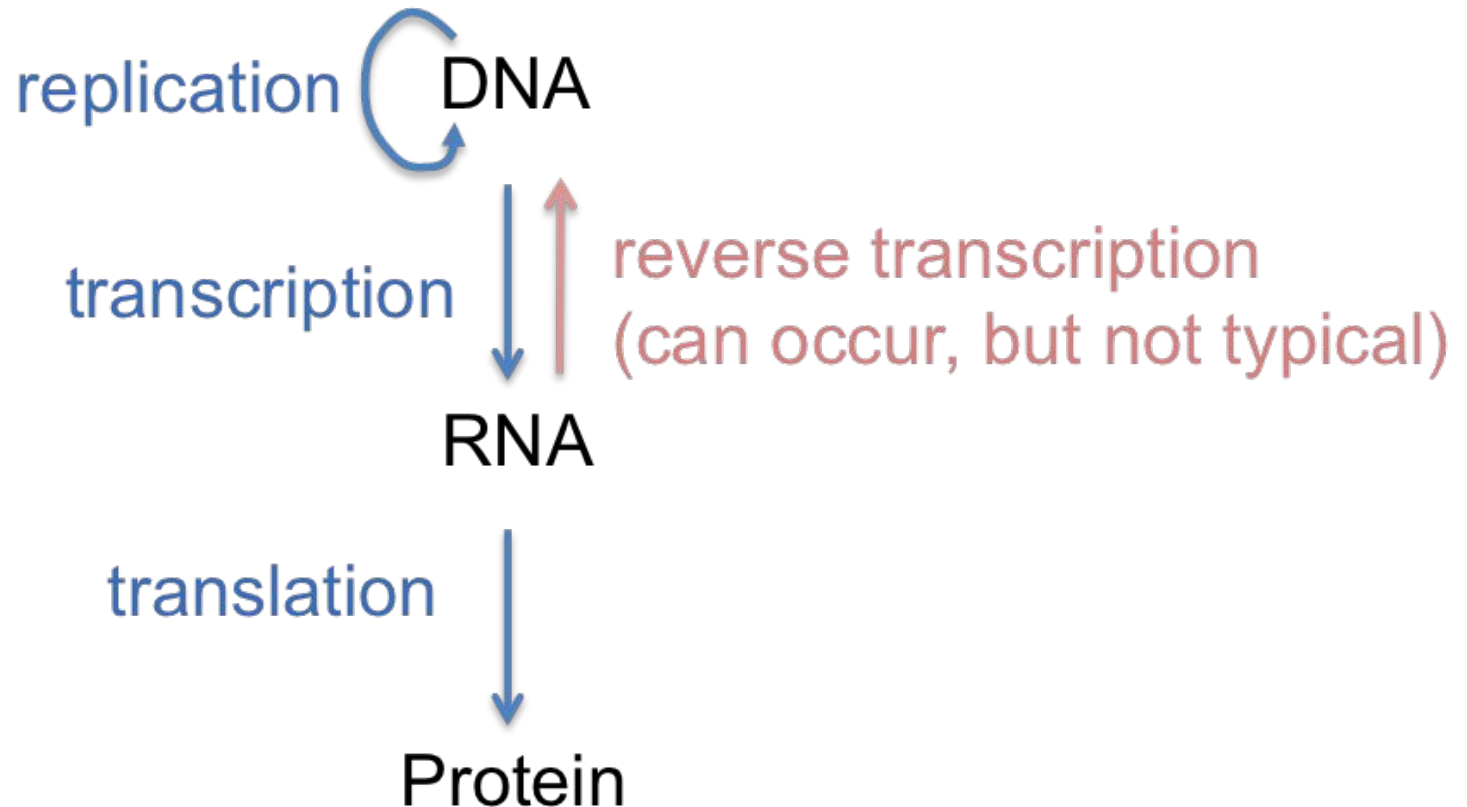
Assignments need to be submitted on Canvas!

# Learning Objectives

Students should be able to:

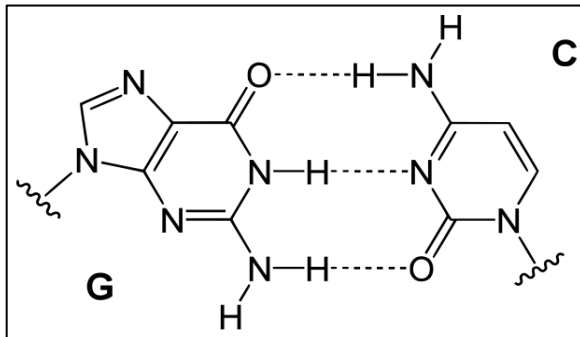
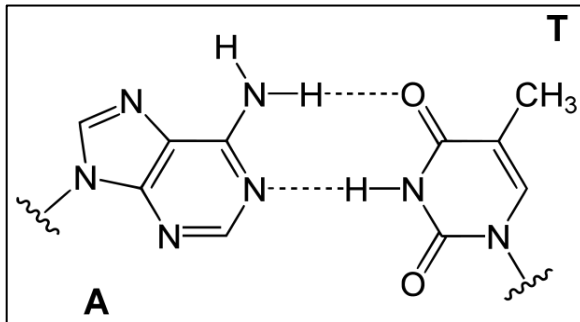
- Explain the Central Dogma and what it means for a gene to be expressed
- Describe what a complementary DNA sequence is
- Recognize conventions for designating DNA sequence directionality
- Print a complementary DNA sequence using "for loops" and "if" statements
- Find the index of characters in a string variable and slice a string
- Use string slicing to make a substitution, deletion, insertion or inversion in a DNA, RNA or protein sequence
- Write out a complementary DNA sequence to a file
- Write out the RNA transcription product for a DNA sequence.
- Define RNA splicing, exon and intron
- Use Python to find possible translation start and stop codons in a mRNA sequence

# Flow of information in biological systems: The Central Dogma



# What is a complementary DNA sequence?

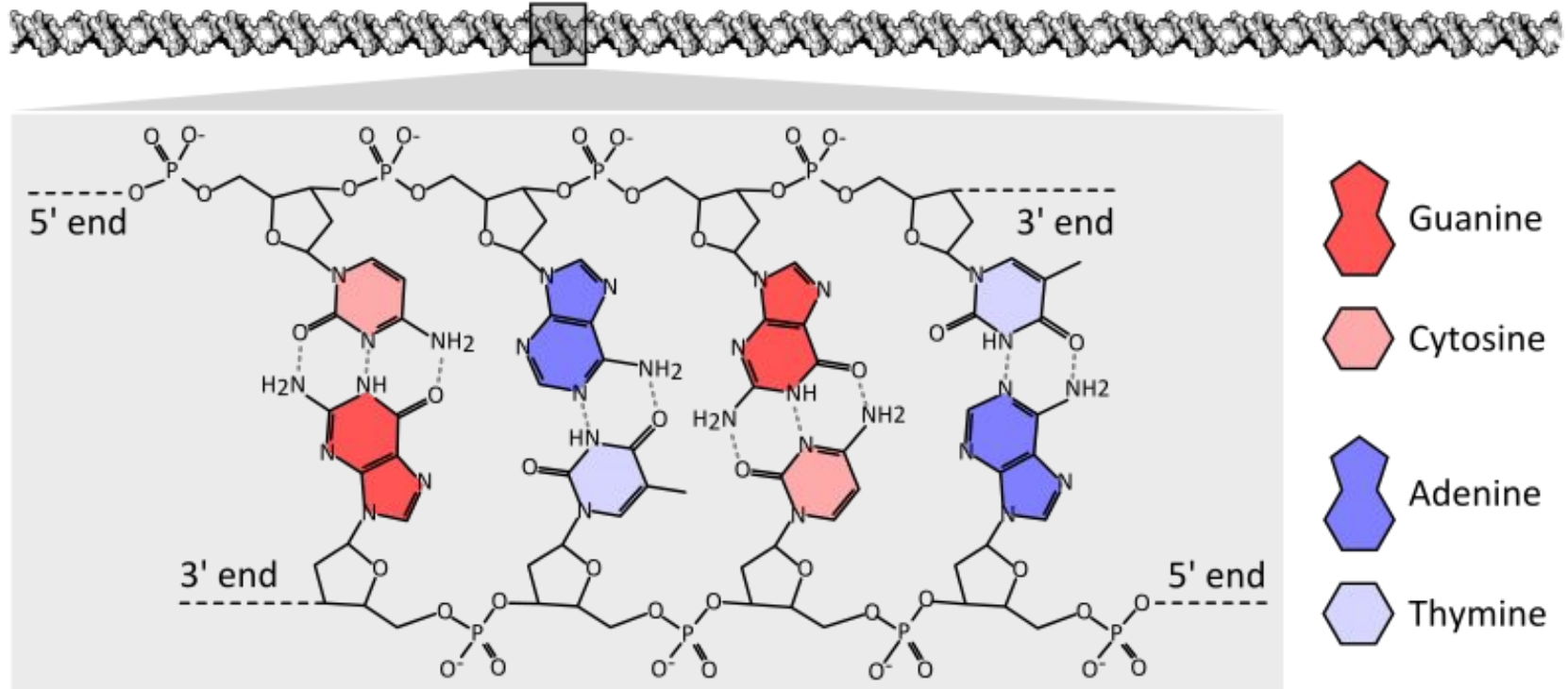
## DNA Base Pairing



Take a moment to write out the complementary DNA sequence for:

AGCCCTCCA

# DNA sequences have directionality



By convention, DNA sequences are written 5' to 3'

Back to our example:

5'-AGCCCTCCA-3' original sequence

3'-TCGGGAGGT-5' complementary strand

5'- -3'

By convention, DNA sequences are  
written 5' to 3'

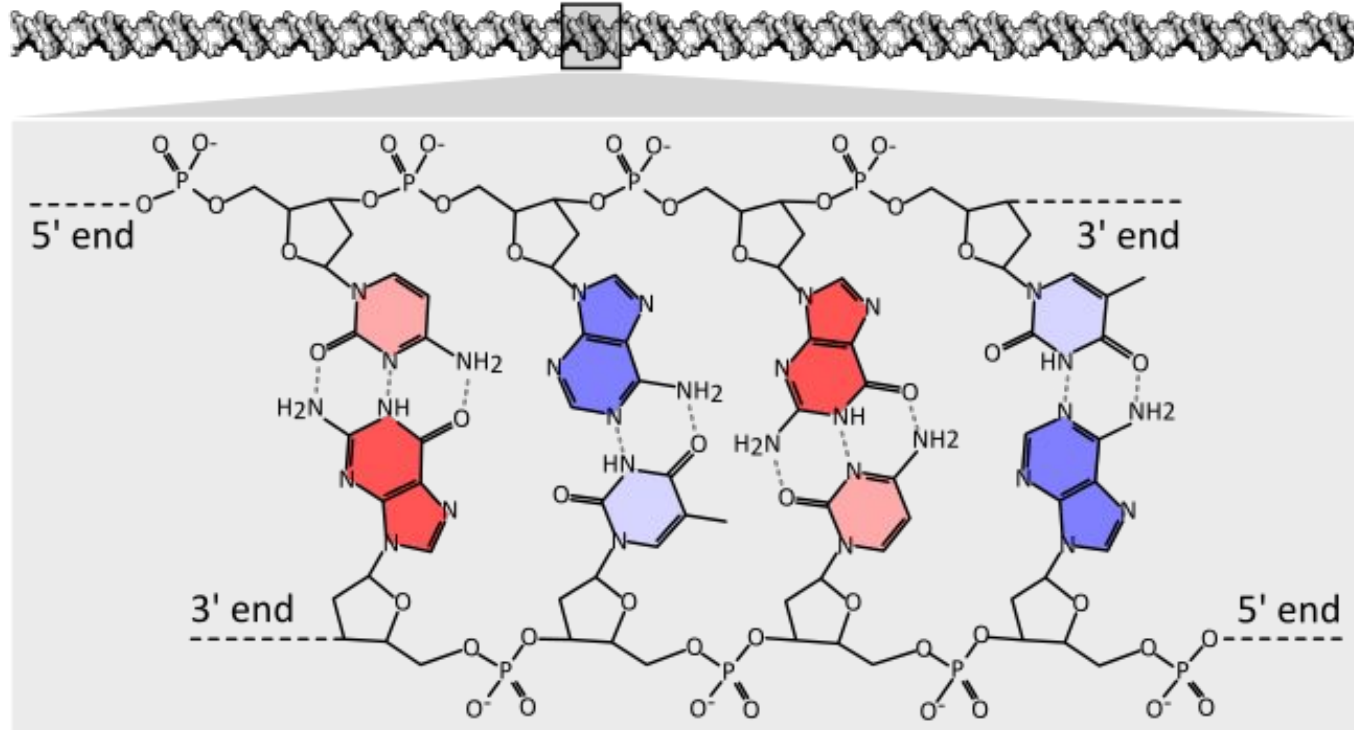
Back to our example:

5'-AGCCCTCCA-3' original sequence

3'-TCGGGAGGT-5' complementary strand

5'-TGGAGGGCT-3' reverse complement

# DNA strands may be referred to as the + or - strand



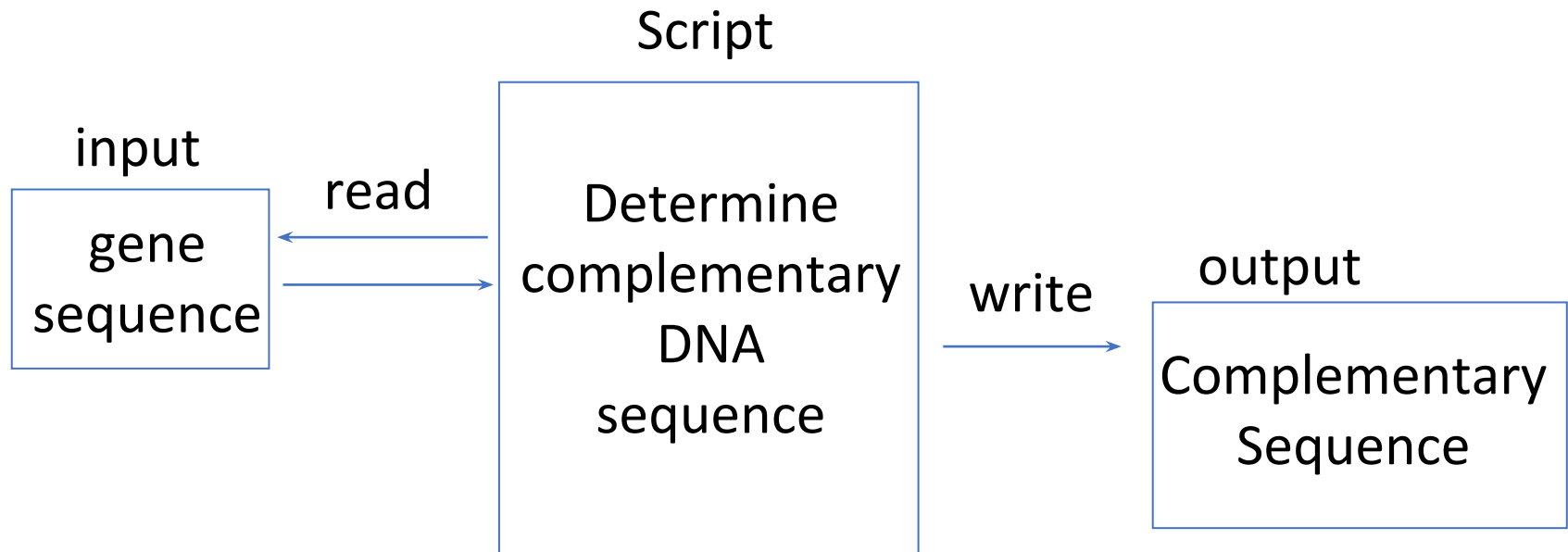
+ strand

- strand



Goal 1: Print a complementary sequence  
using for loops and if statements

# Overview of the script we will write



# What is a for loop?

---

```
In [1]: for i in 'AGCCCTCCA':  
        print (i)
```

# What is a for loop?

---

```
In [1]: for i in 'AGCCCTCCA':  
        print (i)
```

A  
G  
C  
C  
C  
T  
C  
C  
A

How might we use a for loop to write out the complementary sequence?

---

```
In [1]: for i in 'AGCCCTCCA':
```

# How might we use a for loop to write out a complementary sequence?

```
In [2]: #Write out the complementary sequence for a DNA sequence  
for i in 'AGCCCTCCA':  
    if i=='A':  
        print ('T')  
    if i=='T':  
        print ('A')  
    if i=='G':  
        print ('C')  
    if i=='C':  
        print ('G')
```

What if we wanted to write out the reverse complement?

# String slicing

**String:** variables composed of text or characters and are not numbers that can be multiplied, added or divided. Strings are often letters but may have numbers or special characters.

**Indexing** in Python starts with 0

```
In [4]: sequence='TCGGGAGGT'
print (sequence[0])
print (sequence[0:2])
print (sequence[0:3])
print (sequence[1])
print (sequence[1:3])
print (sequence[:4])
```

# String slicing

```
In [4]: sequence='TCGGGAGGT'  
print (sequence[0])  
print (sequence[0:2])  
print (sequence[0:3])  
print (sequence[1])  
print (sequence[1:3])  
print (sequence[:4])
```

T  
TC  
TCG  
C  
CG  
TCGG



# String slicing skipping characters

Adding a stride after the upper bound allows you to skip characters

```
In [5]: sequence='TCGGGAGGT'  
  
#slice the sequence string from [lowerbound:upperbound:stride]  
print (sequence[2:5:2])
```

↑  
Stride

# String slicing skipping characters

```
In [5]: sequence='TCGGGAGGT'

#slice the sequence string from [lowerbound:upperbound:stride]
print (sequence[2:5:2])

GG
```

If a lowerbound is not included slicing occurs at zero.  
If an upper bound is not included slicing occurs at the end of the string.

```
In [6]: sequence='TCGGGAGGT'

#slice the sequence string from [lowerbound:upperbound:stride]
print (sequence[0:9:2])
print (sequence[::2])

TGGGT
TGGGT
```

# A negative stride can reverse a string

```
In [7]: sequence='TCGGGAGGT'  
  
#slice the sequence string from [lowerbound:upperbound:stride] a negative value for the stride reverses the direction.  
print(sequence[::-1])
```

# A negative stride can reverse a string

```
In [7]: sequence='TCGGGAGGT'

#slice the sequence string from [lowerbound:upperbound:stride] a negative value for the stride reverses the direction.
print(sequence[::-1])

TGGAGGGCT
```

# A negative stride can reverse a string

```
In [7]: sequence='TCGGGAGGT'

#slice the sequence string from [lowerbound:upperbound:stride] a negative value for the stride reverses the direction.
print(sequence[::-1])

TGGAGGGCT
```

# String slicing can be used to change (or mutate) parts of a sequence

## Substitution

```
In [8]: #Make a single substitution in a sequence  
sequence='TCGGGAGGT'  
mutated_sequence= sequence[0:4] + 'A' + sequence[6:]  
print(mutated_sequence)
```

TCGGAGGT

# How would you make a:

## Deletion

```
In [9]: #Delete part of a sequence  
sequence='TCGGGAGGT'  
mutated_sequence=  
print(mutated_sequence)
```

TCGGGGT

## Insertion

```
In [10]: #Add an insertion in a sequence  
sequence='TCGGGAGGT'  
mutated_sequence=  
print(mutated_sequence)
```

TCGGATGGT

## Inversion

```
In [12]: #Add an inversion in a sequence  
sequence='TCGGGAGGT'  
print(sequence[7:4:-1])  
mutated_sequence=  
print(mutated_sequence)
```

GGA

TCGGGGGAT

# How would you make a:

## Deletion

```
In [9]: #Delete part of a sequence  
sequence='TCGGGAGGT'  
mutated_sequence= sequence[0:4] + sequence[6:]  
print(mutated_sequence)
```

TCGGGGT

## Insertion

```
In [10]: #Add an insertion in a sequence  
sequence='TCGGGAGGT'  
mutated_sequence= sequence[0:4] + 'AT' + sequence[6:]  
print(mutated_sequence)
```

TCGGATGGT

## Inversion

```
In [12]: #Add an inversion in a sequence  
sequence='TCGGGAGGT'  
print(sequence[7:4:-1])  
mutated_sequence= sequence[0:5] + sequence[7:4:-1] + sequence[8:]  
print(mutated_sequence)
```

GGA

TCGGGGGAT



Substitutions, deletions, insertions and inversions are all types of genetic variation

# How can string slicing be used to write out the reverse complement for a sequence?

```
In [11]: #Write out the complementary sequence for a DNA sequence  
complementarysequence='' #this defines the variable 'complementarysequence'  
for i in 'AGCCCTCCA':  
    if i=='A':  
        complementarysequence=complementarysequence+'T'  
    if i=='T':  
        complementarysequence=complementarysequence+'A'  
    if i=='G':  
        complementarysequence=complementarysequence+'C'  
    if i=='C':  
        complementarysequence=complementarysequence+'G'  
print (complementarysequence)
```

# How can string slicing be used to write out the reverse complement for a sequence?

```
In [11]: #Write out the complementary sequence for a DNA sequence  
complementarysequence='' #this defines the variable 'complementarysequence'  
for i in 'AGCCCTCCA':  
    if i=='A':  
        complementarysequence=complementarysequence+'T'  
    if i=='T':  
        complementarysequence=complementarysequence+'A'  
    if i=='G':  
        complementarysequence=complementarysequence+'C'  
    if i=='C':  
        complementarysequence=complementarysequence+'G'  
print (complementarysequence)  
  
#ANSWER REMOVE BEFORE GIVING TO STUDENTS.  
print (complementarysequence[::-1])
```

```
TCGGGAGGT  
TGGAGGGCT
```

# How can you write a string variable in Python to a file?

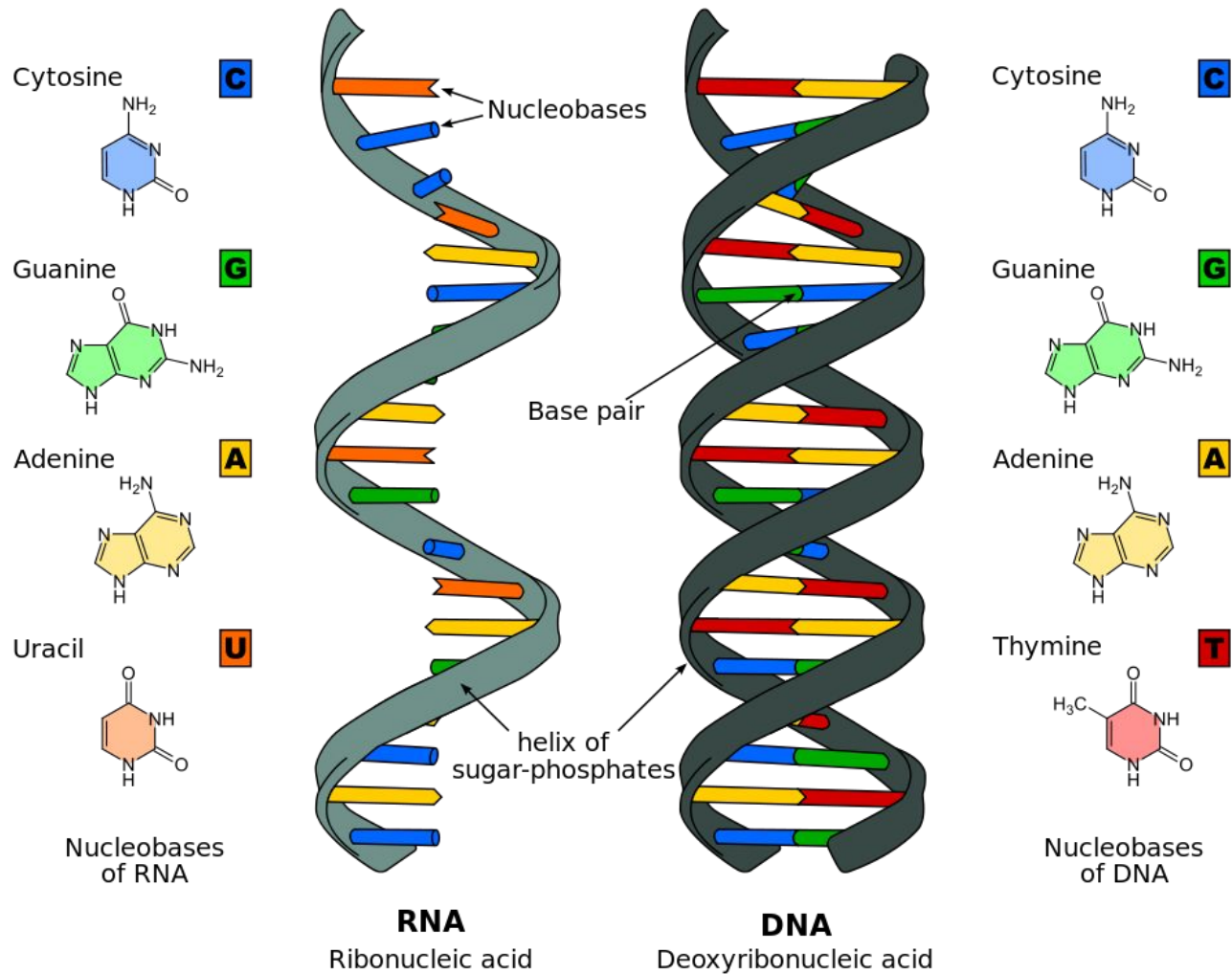
```
#Write out the complementary sequence for a DNA sequence to a file
complementarysequence='' #this defines the variable 'complementarysequence'
for i in 'AGCCCTCCA':
    if i=='A':
        complementarysequence=complementarysequence+'T'
    if i=='T':
        complementarysequence=complementarysequence+'A'
    if i=='G':
        complementarysequence=complementarysequence+'C'
    if i=='C':
        complementarysequence=complementarysequence+'G'

#create a file object f and open a writeable file called 'complementarysequence' in the working directory
f =open('complementarysequence', 'w')

#write the complementary sequence variable to the 'complementarysequence' file
f.write(complementarysequence[::-1])

#close the file object f so it does not take resources in the program
f.close ()
```

# DNA vs RNA



# How can you write the RNA transcription product for a DNA sequence?

```
In [17]: #Write out the pre-mRNA sequence for a DNA sequence
FASTAgenesecquence=open('../class_01_gene_sequences/data/Human-Insulin-NG_007114.1.txt','r')
genesecquence=(FASTAgenesecquence.readlines()[1:])
genesecquence=''.join(genesecquence)
genesecquence=genesecquence.replace('\n','')
RNAsequence='' #this defines the variable 'RNAsequence'

#loops over each character in genesecquence and converts Ts to Us in the RNAsequence variable.
for i in genesecquence:
    if i=='A':
        RNAsequence=RNAsequence+'A'
    if i=='T':
        RNAsequence=RNAsequence+'U'
    if i=='G':
        RNAsequence=RNAsequence+'G'
    if i=='C':
        RNAsequence=RNAsequence+'C'
print ( )
```



# How can you write the RNA transcription product for a DNA sequence?

```
#Write out the pre-mRNA sequence for a DNA sequence
FASTAgeneSequence=open('../class_01_gene_sequences/data/Human-Insulin-NG_007114.1.txt','r')
geneSequence=(FASTAgeneSequence.readlines()[1:])
geneSequence=''.join(geneSequence)
geneSequence=geneSequence.replace('\n','')
RNASequence='' #this defines the variable 'RNASequence'
for i in geneSequence:
    if i=='A':
        RNASequence=RNASequence+'A'
    if i=='T':
        RNASequence=RNASequence+'U'
    if i=='G':
        RNASequence=RNASequence+'G'
    if i=='C':
        RNASequence=RNASequence+'C'
print (RNASequence)
```

# How can you write the RNA transcription product for a DNA sequence using if/else statements?

```
In [18]: #Transcription: Writes out the pre-mRNA sequence that will be made from a DNA sequence
FASTAgenesequence=open('../class_01_gene_sequences/data/Human-Insulin-NG_007114.1.txt','r')
genesequence=(FASTAgenesequence.readlines()[1:])
genesequence=''.join(genesequence)
genesequence=genesequence.replace('\n','')
RNAsequence='' #this defines the variable 'RNAsequence'

#loops over each character in genesequence and converts Ts to Us in the RNAsequence variable.
for i in genesequence:
    if i==' ':
        RNAsequence=RNAsequence+' '
    else:
        RNAsequence=RNAsequence+ i
print ( )
```



# How can you write the RNA transcription product for a DNA sequence using if/else statements?

```
##ANSWER -- REMOVE BEFORE GIVING TO STUDENTS ##  
#Write out the pre-mRNA sequence for a DNA sequence  
FASTAgeneSequence=open('../class_01_gene_sequences/data/Human-Insulin-NG_007114.1.txt','r')  
geneSequence=(FASTAgeneSequence.readlines()[1:])  
geneSequence=''.join(geneSequence)  
RNASequence='' #this defines the variable 'RNASequence'  
for i in geneSequence:  
    if i=='T':  
        RNASequence=RNASequence+'U'  
    else:  
        RNASequence=RNASequence+ i  
print (RNASequence)
```

```
AGCCCUCCAGGACAGGCUGCAUCAGAAGAGGCCAUCAAGCAGGUCUGUCCAAGGGCCUUUGCGUCAGGU  
GGGCUCAGGAUCCAGGGUGGCUGGACCCAGGCCCCAGCUCUGCAGCAGGGAGGACGUGGCUGGGCUCG  
UGAAGCAUGUGGGGGUGAGCCCAGGGGCCCAAGGCAGGGCACCUGGCCUUCAGCCUGCCUCAGCCCUGC
```

What are RNA splicing, exons and introns?

# Using Python to find start and stop codons in an mRNA sequence

```
In [23]: #Find possible start codons in an mRNA sequence
FASTAmRNAsequence=open('../class_01_gene_sequences/data/Human-Insulin NM_000207.2.txt','r')
mRNAsequence=(FASTAmRNAsequence.readlines()[1:])
mRNAsequence=''.join(mRNAsequence)
mRNAsequence=mRNAsequence.replace('\n','')

#loops over every set of three consecutive basepairs in the mRNAsequence and looks to see if it is
an ATG.
for i in range(0,len(mRNAsequence)):
    if mRNAsequence[i:i+3]=='ATG':
#prints the possible start codon starting position, the i+1 converts the start position to 1.
        print('candidate start codon site: '+ str(i+1))

candidate start codon site: 60
candidate start codon site: 72
candidate start codon site: 341
candidate start codon site: 442
```