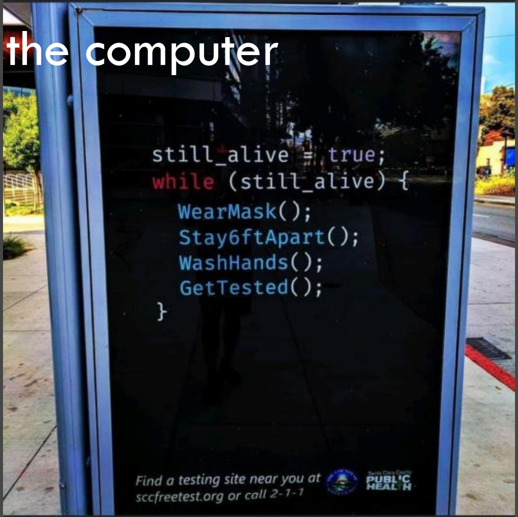
SESSION 2.

Working with the molecules of life in the computer



```
still_alive = true;
while (still_alive) {
   WearMask();
   Stay6ftApart();
   WashHands();
   GetTested();
}
```

Find a testing site near you at sccfreetest.org or call 2-1-1

Counts in Biology

- How many RNA copies are in an average mammalian cell?
- How many somatic or de novo mutations are per a cell?
- How many nucleotides are different between siblings and friends?
- How much DNA are similar between human and fly?
- How many DNA are created in my body per day?

- Count questions from you!
- 하루에 세포가 몇개가 내 몸에서 죽는지?
- 면역반응에서 몇개의 유전자의 발현이 증가 할 것이가?
- 하루밤 사이에 새롭게 만들어지는 세포의 개수
- 엄마와 내가 DNA methylation site가 몇개가 다른가?

What is life? Evolution - Mutation

- Life (object) is made of “matter”, “energy”, and “information (or data)”.

□ Life’s forms are very different but their basic genetic information is very similar from bacteria to human. → Evolution

□ Why the information is similar?

□ All species are related and have a common ancestor (Mutation and selection on DNAs made a process of the evolution)

□ The concept of evolution is fundamental in genomics and bioinformatics.

Discrete information of DNA (bases) is computable.

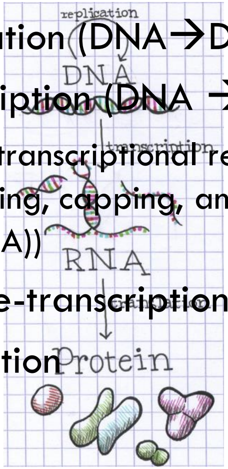
- □ Mutations and inheritance are measurable to a discrete value.
- □ Genetic materials are not computable but the genetic information is computable.
- □ Homology of genes is not computable but the similarity is computable.

- □ Similarity links the DNA sequences to a biological function.

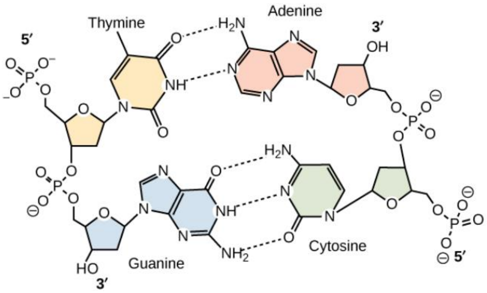
- Data vs Information
  - ➢ Expected lifespan of males vs females
  - ➢ Mutation profiles of smokers vs nonsmokers

Flow of genetic information in a eukaryotic cell

- DNA (deoxyribonucleic acid)
- Central dogma
  - Replication (DNA→DNA)
  - Transcription (DNA → RNA)
    - Post-transcriptional reg. (splicing, capping, and poly(A))
  - Reverse-transcription
  - Translation



- RNA and protein sequences are also discrete values.
- DNA has a chemical polarity (5'→3') and antiparallel.
- Base-complementarity (C,T=G,A)



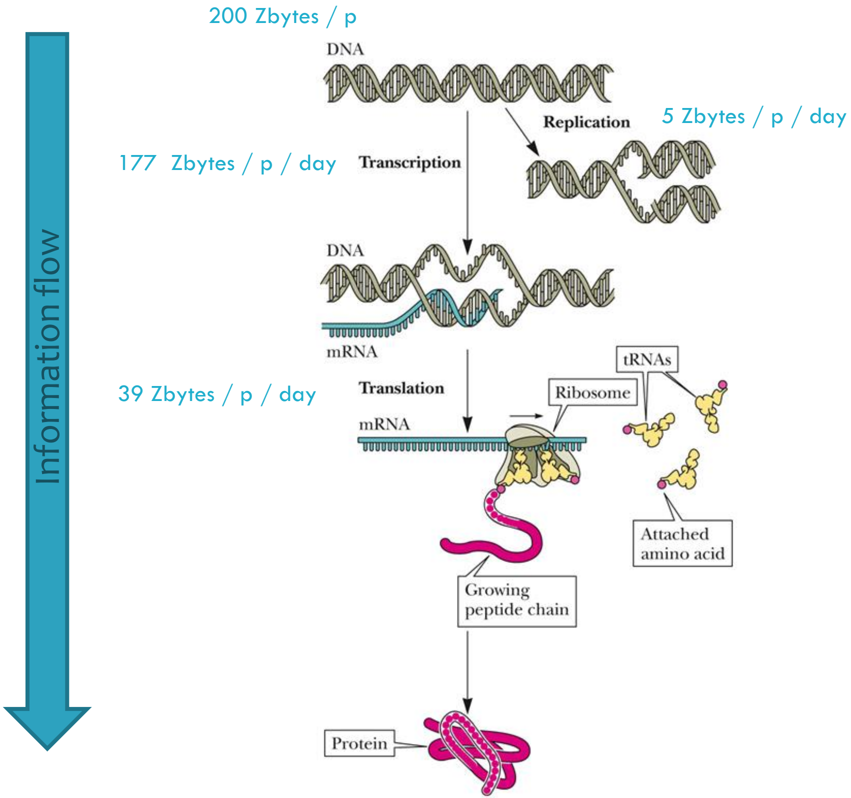
Genetic information that flows along molecular cascades is big data

DNA
(ACGT)

RNA
(ACGU)

AA
(20 aa)

Information flow



200 Zbytes / p

5 Zbytes / p / day

177 Zbytes / p / day

39 Zbytes / p / day

Flow of genetic information in a eukaryotic cell

- Gene structure – Exons (5'UTR, CDS, 3'UTR) and introns.
- Mature mRNA – 5'UTR, CDS, 3'UTR, and Poly(A)
- Single-stranded DNA sequences
  - `5'-AGGACACGACGACTATTGG-3'`
- Double-stranded DNA sequences
  - `5'-AGGACACGACGACTATTGG-3'`
  - `3'-TCCTGTGCTGCTGATAACC-5'`

- Forward/Reverse, +/-, Watson/Crick, sense/antisense
- Save a ssDNA form in database due to interpretability.

Replication (DNA→DNA)

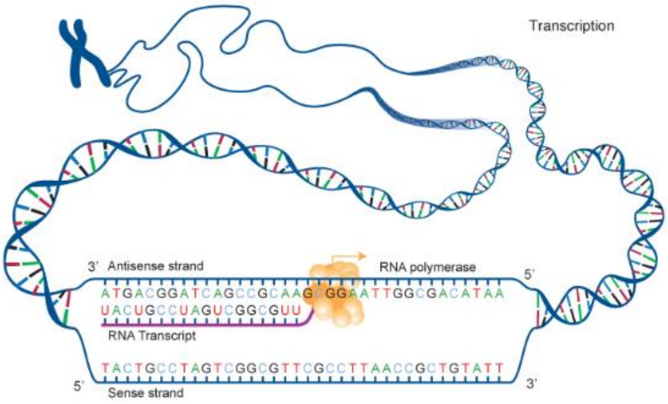
- 22 autosomes and Two sex chromosomes (X,Y)
- The human genome – genetic information in all chromosomes (3 billion bases).

```
5’-AGGACACG →              -3’
3’-TCCTGTGCTGCTGATAACC-5’
```

- During replication, the erroneous NT incorporation occurs but the error correction system greatly reduce the mutations.
- Each replication creates 1-2 mutations in average.
- The replication process is to synthesize one strand using the other strand as a template.

Transcription (DNA → RNA)

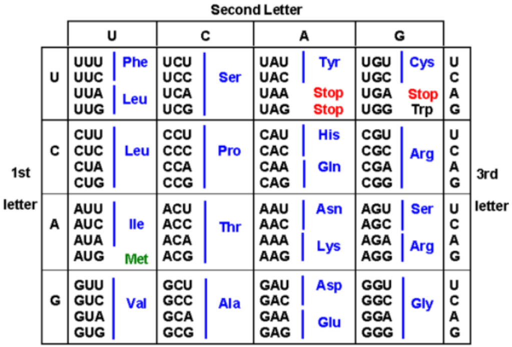
- RNA polymerase uses antisense strand as a template DNA and RNA itself is same as the sense.
- DNA→RNA : T to U



- RNA is a single strand.
- These RNAs contain information for the production of proteins.

Translation

- RNA to Protein
- Proteins are large polymers like DNA and RNA, the building blocks are 20 amino acids (string-like, discrete).
- A distinct aa is defined by a codon (three bases).
- There are 64 codons (61 specify aa and 3 define stop codons).
- Multiple codons code an aa (codon degeneracy)
- AUG: start codon and codes methionine.
- tRNAs and aminoacyl transferases link the codons and amino acids.



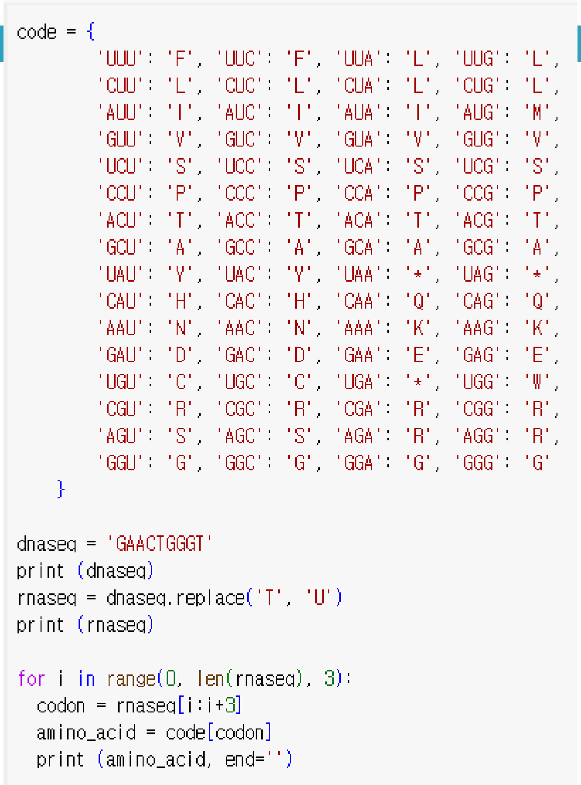
Second Letter

1st letter	U		C		A		G		3rd letter
U	UUU	Phe	UCU	Ser	UAU	Tyr	UGU	Cys	U
	UUC		UCC		UAC		UGC		C
	UUA	Leu	UCA		UAA	Stop	UGA	Stop	A
	UUG		UCG		UAG	Stop	UGG	Trp	G
C	CUU	Leu	CCU	Pro	CAU	His	CGU	Arg	U
	CUC		CCC		CAC		CGC		C
	CUA		CCA		CAA	Gln	CGA		A
	CUG		CCG		CAG		CGG		G
A	AUU	Ile	ACU	Thr	AAU	Asn	AGU	Ser	U
	AUC		ACC		AAC		AGC		C
	AUA		ACA		AAA	Lys	AGA	Arg	A
	AUG	Met	ACG		AAG		AGG		G
G	GUU	Val	GCU	Ala	GAU	Asp	GGU	Gly	U
	GUC		GCC		GAC		GGC		C
	GUA		GCA		GAA	Glu	GGA		A
	GUG		GCG		GAG		GGG		G

Python program mimicking the genetic information flow

- □ Python
  - Basic syntax (grammar)
  - Defining variables
  - Print, reverse, string functions
  - For loops
  - Dictionary (hash table)

- □ Genetic information flow

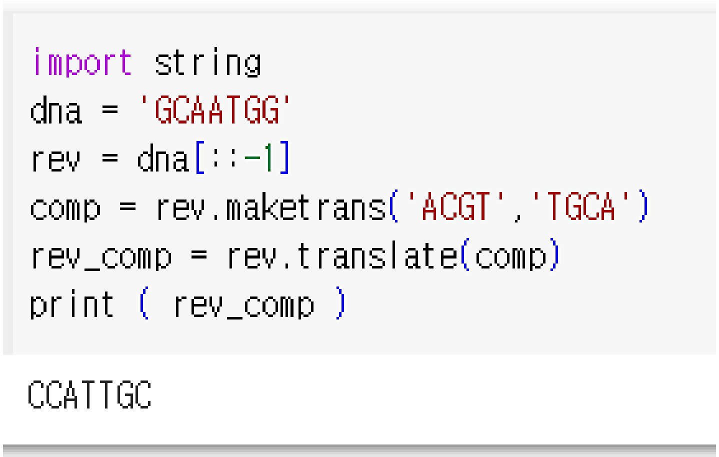


```
code = {
        'UUU': 'F', 'UUC': 'F', 'UUA': 'L', 'UUG': 'L',
        'CUU': 'L', 'CUC': 'L', 'CUA': 'L', 'CUG': 'L',
        'AUU': 'I', 'AUC': 'I', 'AUA': 'I', 'AUG': 'M',
        'GUU': 'V', 'GUC': 'V', 'GUA': 'V', 'GUG': 'V',
        'UCU': 'S', 'UCC': 'S', 'UCA': 'S', 'UCG': 'S',
        'CCU': 'P', 'CCC': 'P', 'CCA': 'P', 'CCG': 'P',
        'ACU': 'T', 'ACC': 'T', 'ACA': 'T', 'ACG': 'T',
        'GCU': 'A', 'GCC': 'A', 'GCA': 'A', 'GCG': 'A',
        'UAU': 'Y', 'UAC': 'Y', 'UAA': '*', 'UAG': '*',
        'CAU': 'H', 'CAC': 'H', 'CAA': 'Q', 'CAG': 'Q',
        'AAU': 'N', 'AAC': 'N', 'AAA': 'K', 'AAG': 'K',
        'GAU': 'D', 'GAC': 'D', 'GAA': 'E', 'GAG': 'E',
        'UGU': 'C', 'UGC': 'C', 'UGA': '*', 'UGG': 'W',
        'CGU': 'R', 'CGC': 'R', 'CGA': 'R', 'CGG': 'R',
        'AGU': 'S', 'AGC': 'S', 'AGA': 'R', 'AGG': 'R',
        'GGU': 'G', 'GGC': 'G', 'GGA': 'G', 'GGG': 'G'
    }

dnaseq = 'GAACTGGGT'
print (dnaseq)
rnaseq = dnaseq.replace('T', 'U')
print (rnaseq)

for i in range(0, len(rnaseq), 3):
  codon = rnaseq[i:i+3]
  amino_acid = code[codon]
  print (amino_acid, end='')
```

DNA replication I (ssDNA)



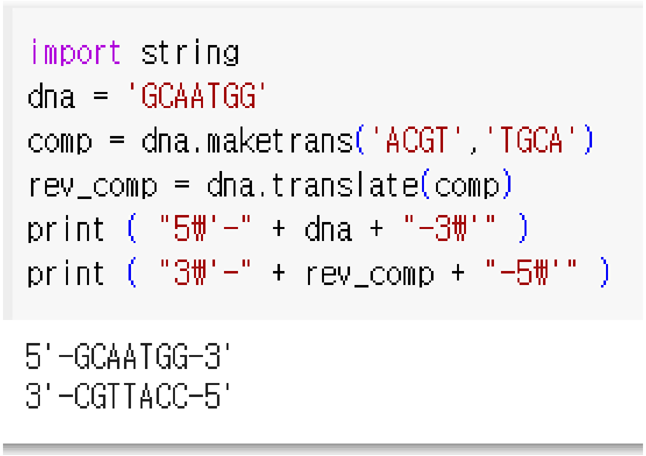
```
import string
dna = 'GCAATGG'
rev = dna[::-1]
comp = rev.maketrans('ACGT','TGCA')
rev_comp = rev.translate(comp)
print ( rev_comp )
```

```
CCATTGC
```

Write the above code into a file “replication.py”

And run like “python replication.py”

DNA replication II (dsDNA)



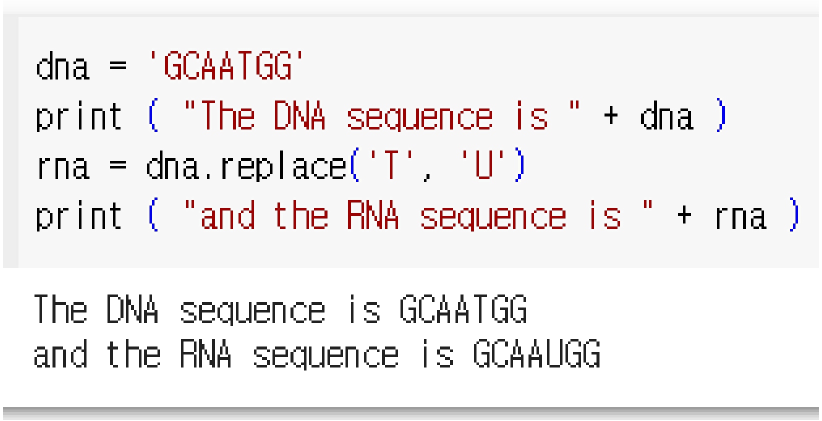
```
import string
dna = 'GCAATGG'
comp = dna.maketrans('ACGT','TGCA')
rev_comp = dna.translate(comp)
print ( "5₩'-" + dna + "-3₩'" )
print ( "3₩'-" + rev_comp + "-5₩'" )
```

```
5'-GCAATGG-3'
3'-CGTTACC-5'
```

Write the above code into a file "replication2.py"

And run like "python replication2.py"

Inferring RNA products of transcription



```
dna = 'GCAATGG'
print ( "The DNA sequence is " + dna )
rna = dna.replace('T', 'U')
print ( "and the RNA sequence is " + rna )
```

```
The DNA sequence is GCAATGG
and the RNA sequence is GCAAUGG
```

Write the above code into a file "transcription.py"
And run like "python transcription.py"

Inferring protein products of translation

- □ codons → amino acids and stop codons using a dictionary (hash table)

```
code[‘UUU’]=‘F’
code[‘UUC’]=‘F’
…
code[‘GGG’]=‘G’
```

→

```
code = {‘UUU’:’F’, ‘UUC’:’F’, ‘UUA’:’L’, … }
print (code[‘UUU’], code[‘UUC’])
```

```
F F
```

Inferring protein products of translation

- Reading the codon from the RNA sequence using a for loop statement.

```
for i in range(0, 5, 1):
        print (i, end=‘ ’)
```

```
0 1 2 3 4
```

Inferring protein products of translation

- □ Subtract codons (substring) from a RNA sequence (string)

```
rna='AGCTT'
print (rna[2:4])
print (rna[0:1])
print (rna[3:])
print (rna[:3])
print (rna[:-2])
print (rna[::-1])
```

```
CT
A
TT
AGC
AGC
TTCGA
```

code1.1 translation.py

```
#!/usr/bin/python
code = {
    'UUU': 'F', 'UUC': 'F', 'UUA': 'L', 'UUG': 'L', 'CUU': 'L', 'CUC': 'L', 'CUA': 'L', 'CUG': 'L',
    'AUU': 'I', 'AUC': 'I', 'AUA': 'I', 'AUG': 'M', 'GUU': 'V', 'GUC': 'V', 'GUA': 'V', 'GUG': 'V',
    'UCU': 'S', 'UCC': 'S', 'UCA': 'S', 'UCG': 'S', 'CCU': 'P', 'CCC': 'P', 'CCA': 'P', 'CCG': 'P',
    'ACU': 'T', 'ACC': 'T', 'ACA': 'T', 'ACG': 'T', 'GCU': 'A', 'GCC': 'A', 'GCA': 'A', 'GCG': 'A',
    'UAU': 'Y', 'UAC': 'Y', 'UAA': '*', 'UAG': '*', 'CAU': 'H', 'CAC': 'H', 'CAA': 'Q', 'CAG': 'Q',
    'AAU': 'N', 'AAC': 'N', 'AAA': 'K', 'AAG': 'K', 'GAU': 'D', 'GAC': 'D', 'GAA': 'E', 'GAG': 'E',
    'UGU': 'C', 'UGC': 'C', 'UGA': '*', 'UGG': 'W', 'CGU': 'R', 'CGC': 'R', 'CGA': 'R', 'CGG': 'R',
    'AGU': 'S', 'AGC': 'S', 'AGA': 'R', 'AGG': 'R','GGU': 'G', 'GGC': 'G', 'GGA': 'G', 'GGG': 'G'
    }

 dnaseq = 'GAACTGGGT'
 print (dnaseq)
 rnaseq = dnaseq.replace('T', 'U')
 print (rnaseq)

 for i in range(0, len(rnaseq), 3):
     codon = rnaseq[i:i + 3]
     amino_acid = code[codon]
     print (amino_acid, end=' ')
```

```
GAACTGGGT
GAACUGGGU
 E  L  G
```