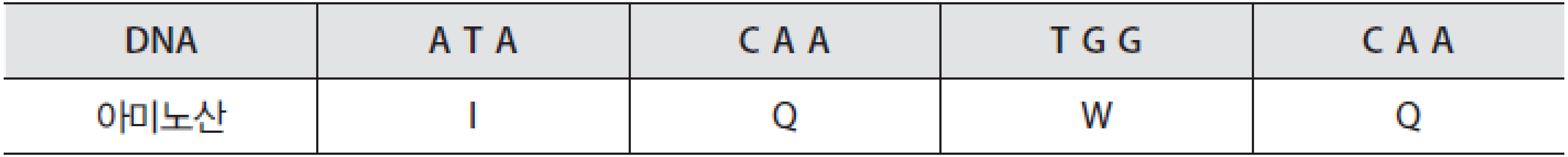
DNA 분석

DNA 분석 과정

DNA 서열을 아미노산의 서열로 변환

- 12개의 알파벳으로 구성되어 있는 DNA 데이터를 3개씩 잘라서 하나의 아미노산으로 변환

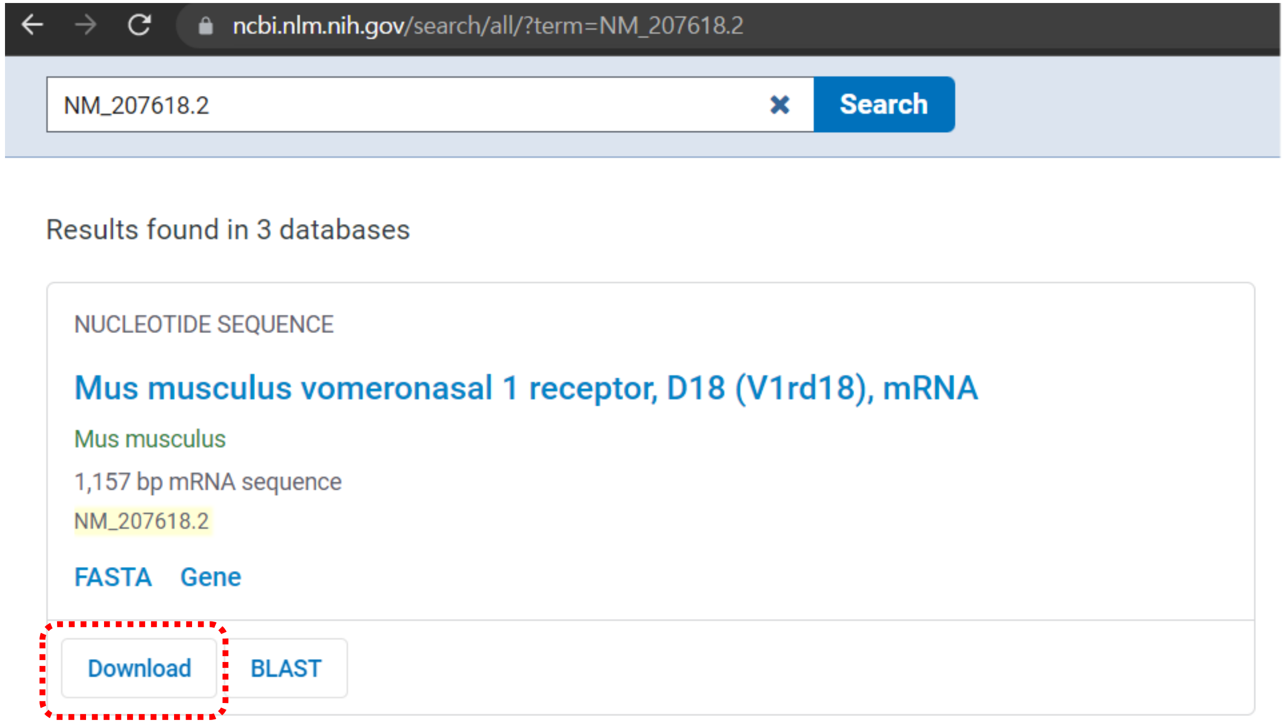


DNA	A T A	C A A	T G G	C A A
아미노산	I	Q	W	Q

데이터 다운로드

DNA와 프로틴 시퀀스 데이터를 수동 다운로드

- NCBI(https://www.ncbi.nlm.nih.gov/) 접속
- NM_207618.2를 검색(search)
- Nucleotide sequence(뉴클레오티드 서열) 다운로드
- "NM_207618.2.fasta"



ncbi.nlm.nih.gov/search/all/?term=NM_207618.2

NM_207618.2 ✖ Search

Results found in 3 databases

NUCLEOTIDE SEQUENCE

Mus musculus vomeronasal 1 receptor, D18 (V1rd18), mRNA

Mus musculus

1,157 bp mRNA sequence

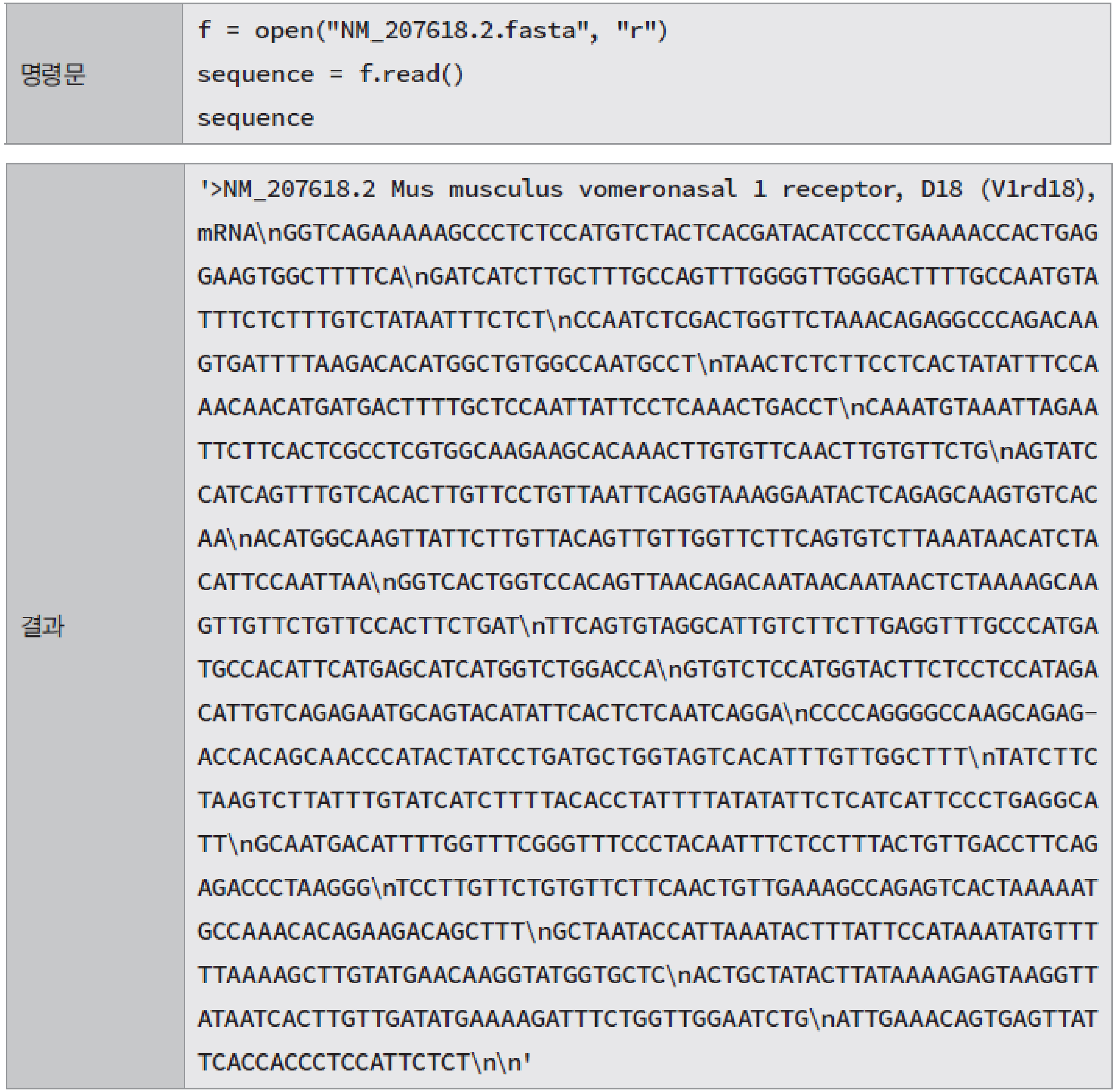
NM_207618.2

FASTA Gene

Download BLAST

파이썬으로 DNA 데이터 가공

- open() 명령을 이용하여 읽기 전용으로 파일 열기
- 내용을 출력



명령문	f = open("NM_207618.2.fasta", "r")
	sequence = f.read()
	sequence

결과	'>NM_207618.2 Mus musculus vomeronasal 1 receptor, D18 (V1rd18), mRNA\nGGTCAGAAAAAGCCCTCTCCATGTCTACTCACGATACATCCCTGAAAACCACTGAG GAAGTGGCTTTTCA\nGATCATCTTGCTTTGCCAGTTTGGGGTTGGGACTTTTGCCAATGTA TTTCTCTTTGTCTATAATTTCTCT\nCCAATCTCGACTGGTTCTAAACAGAGGCCCAGACAA GTGATTTTAAGACACATGGCTGTGGCCAATGCCT\nTAACTCTCTTCCTCACTATATTTCCA AACAACATGATGACTTTTGCTCCAATTATTCCTCAAACTGACCT\nCAAATGTAAATTAGAA TTCTTCACTCGCCTCGTGGCAAGAAGCACAAACTTGTGTTCAACTTGTGTTCTG\nAGTATC CATCAGTTTGTCACACTTGTTCCTGTTAATTCAGGTAAAGGAATACTCAGAGCAAGTGTCAC AA\nACATGGCAAGTTATTCTTGTTACAGTTGTTGGTTCTTCAGTGTCTTAAATAACATCTA CATTCCAATTAA\nGGTCACTGGTCCACAGTTAACAGACAATAACAATAACTCTAAAAGCAA GTTGTTCTGTTCCACTTCTGAT\nTTCAGTGTAGGCATTGTCTTCTTGAGGTTTGCCCATGA TGCCACATTCATGAGCATCATGGTCTGGACCA\nGTGTCTCCATGGTACTTCTCCTCCATAGA CATTGTCAGAGAATGCAGTACATATTCACTCTCAATCAGGA\nCCCCAGGGGCCAAGCAGAG- ACCACAGCAACCCATACTATCCTGATGCTGGTAGTCACATTTGTTGGCTTT\nTATCTTC TAAGTCTTATTTGTATCATCTTTTACACCTATTTTATATATTCTCATCATTCCCTGAGGCA TT\nGCAATGACATTTTGGTTTCGGGTTTCCCTACAATTTCTCCTTTACTGTTGACCTTCAG AGACCCTAAGGG\nTCCTTGTTCTGTGTTCTTCAACTGTTGAAAGCCAGAGTCACTAAAAAT GCCAAACACAGAAGACAGCTTT\nGCTAATACCATTAAATACTTTATTCCATAAATATGTTT TTAAAAGCTTGTATGAACAAGGTATGGTGCTC\nACTGCTATACTTATAAAAGAGTAAGGTT ATAATCACTTGTTGATATGAAAAGATTTCTGGTTGGAATCTG\nATTGAAACAGTGAGTTAT TCACCACCCTCCATTCTCT\n\n'

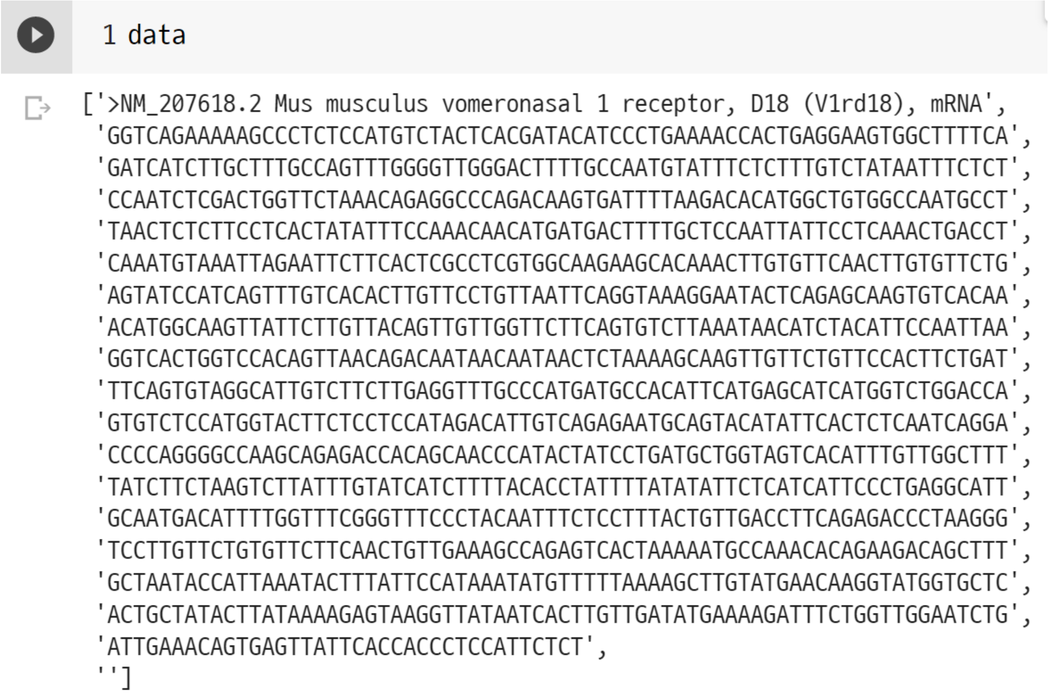
파이썬으로 DNA 데이터 가공

불필요한 행 삭제

- **splitlines()** 이용
- 문자열을 줄바꿈 기호 기준으로 쪼개기
- 아래 data 변수 내용 확인

명령문	with open('NM_2076182.fasta', 'r') as inf:
	    data = inf.read().splitlines()
	with open('dna1.txt', 'w') as outf:
	    outf.writelines(data[1:])
	f = open('dna1.txt', 'r')
	sequence = f.read()
	sequence

결과	'GGTCAGAAAAAGCCCTCTCCATGTCTACTCACGATACATCCCTGAAAACCACTGAGGAAGT GGCTTTTCA\nGATCATCTTGCTTTGCCAGTTTGGGGTTGGGACTTTTGCCAATGTATTTCT CTTTGTCTATAATTTCTCT\nCCAATCTCGACTGGTTCTAAACAGAGGCCCAGACAAGTGAT TTTAAGACACATGGCTGTGGCCAATGCCT\nTAACTCTCTTCCTCACTATATTTCC



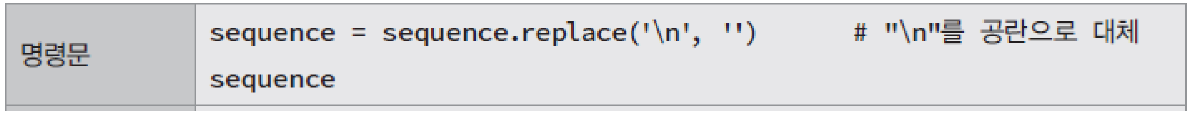
```
1 data
```

```
['>NM_207618.2 Mus musculus vomeronasal 1 receptor, D18 (V1rd18), mRNA',
 'GGTCAGAAAAAGCCCTCTCCATGTCTACTCACGATACATCCCTGAAAACCACTGAGGAAGTGGCTTTTCA',
 'GATCATCTTGCTTTGCCAGTTTGGGGTTGGGACTTTTGCCAATGTATTTCTCTTTGTCTATAATTTCTCT',
 'CCAATCTCGACTGGTTCTAAACAGAGGCCCAGACAAGTGATTTTAAGACACATGGCTGTGGCCAATGCCT',
 'TAACTCTCTTCCTCACTATATTTCCAAACAACATGATGACTTTTGCTCCAATTATTCCTCAAACTGACCT',
 'CAAATGTAAATTAGAATTCTTCACTCGCCTCGTGGCAAGAAGCACAAACTTGTGTTCAACTTGTGTTCTG',
 'AGTATCCATCAGTTTGTCACACTTGTTCCTGTTAATTCAGGTAAAGGAATACTCAGAGCAAGTGTCACAA',
 'ACATGGCAAGTTATTCTTGTTACAGTTGTTGGTTCTTCAGTGTCTTAAATAACATCTACATTCCAATTAA',
 'GGTCACTGGTCCACAGTTAACAGACAATAACAATAACTCTAAAAGCAAGTTGTTCTGTTCCACTTCTGAT',
 'TTCAGTGTAGGCATTGTCTTCTTGAGGTTTGCCCATGATGCCACATTCATGAGCATCATGGTCTGGACCA',
 'GTGTCTCCATGGTACTTCTCCTCCATAGACATTGTCAGAGAATGCAGTACATATTCACTCTCAATCAGGA',
 'CCCCAGGGGCCAAGCAGAGACCACAGCAACCCATACTATCCTGATGCTGGTAGTCACATTTGTTGGCTTT',
 'TATCTTCTAAGTCTTATTTGTATCATCTTTTACACCTATTTTATATATTCTCATCATTCCCTGAGGCATT',
 'GCAATGACATTTTGGTTTCGGGTTTCCCTACAATTTCTCCTTTACTGTTGACCTTCAGAGACCCTAAGGG',
 'TCCTTGTTCTGTGTTCTTCAACTGTTGAAAGCCAGAGTCACTAAAAATGCCAAACACAGAAGACAGCTTT',
 'GCTAATACCATTAAATACTTTATTCCATAAATATGTTTTTAAAAGCTTGTATGAACAAGGTATGGTGCTC',
 'ACTGCTATACTTATAAAAGAGTAAGGTTATAATCACTTGTTGATATGAAAAGATTTCTGGTTGGAATCTG',
 'ATTGAAACAGTGAGTTATTCACCACCCTCCATTCTCT',
 '']
```

파이썬으로 DNA 데이터 가공

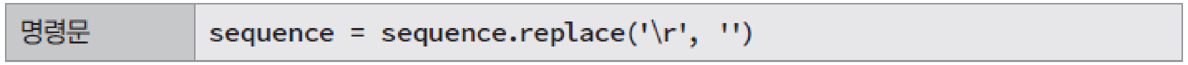
▪ 불필요한 문자 삭제

- '\n' 삭제



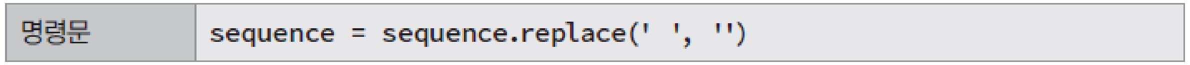
명령문	sequence = sequence.replace('\n', '') # "\n"를 공란으로 대체 sequence

- '\r' 삭제



명령문	sequence = sequence.replace('\r', '')

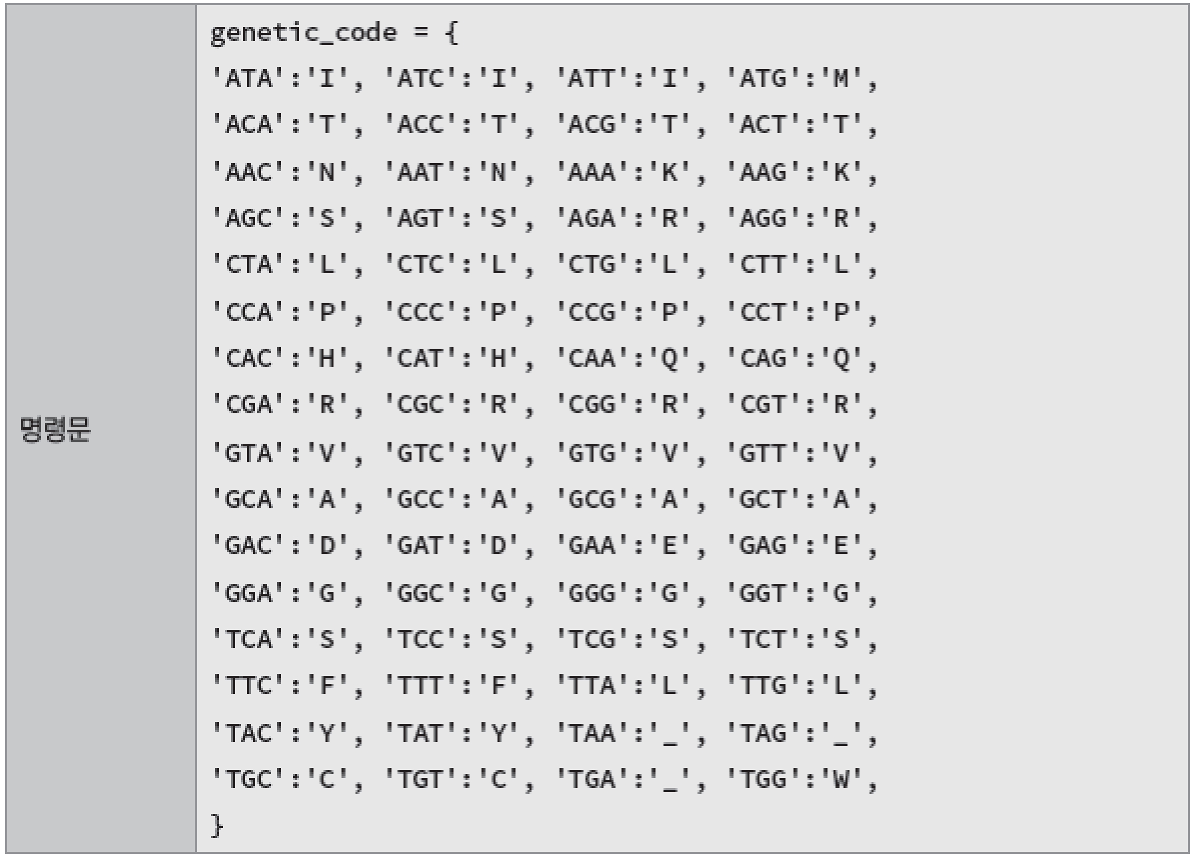
- ' ' 삭제



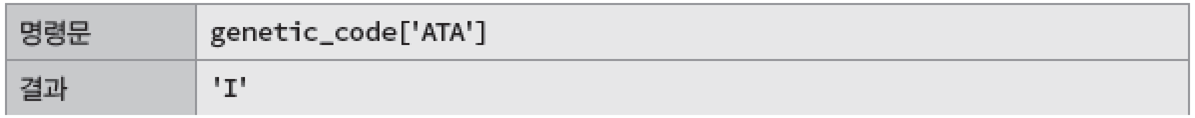
명령문	sequence = sequence.replace(' ', '')

DNA를 아미노산으로 변환

- DNA에 대한 아미노산의 정보



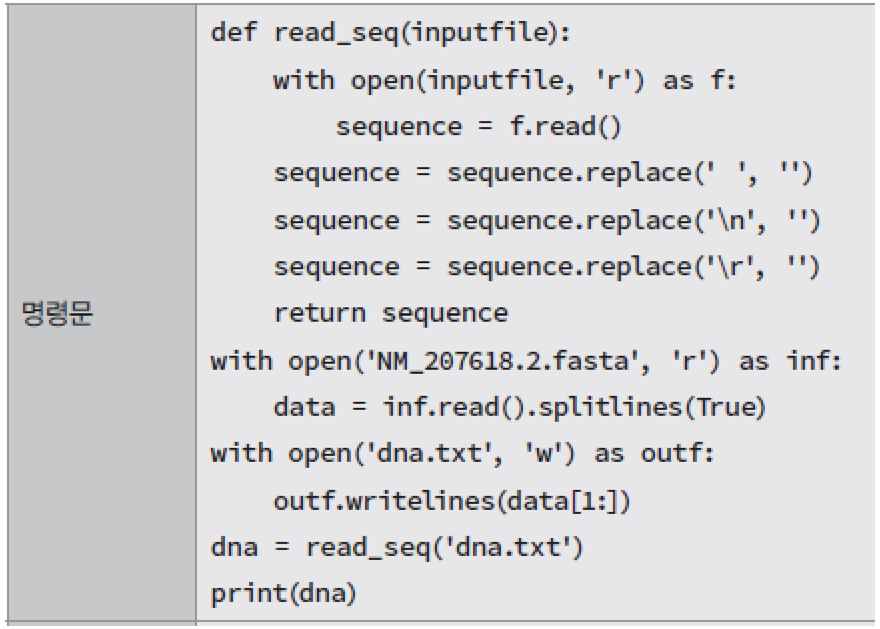
명령문	genetic_code = {
	'ATA':'I', 'ATC':'I', 'ATT':'I', 'ATG':'M',
	'ACA':'T', 'ACC':'T', 'ACG':'T', 'ACT':'T',
	'AAC':'N', 'AAT':'N', 'AAA':'K', 'AAG':'K',
	'AGC':'S', 'AGT':'S', 'AGA':'R', 'AGG':'R',
	'CTA':'L', 'CTC':'L', 'CTG':'L', 'CTT':'L',
	'CCA':'P', 'CCC':'P', 'CCG':'P', 'CCT':'P',
	'CAC':'H', 'CAT':'H', 'CAA':'Q', 'CAG':'Q',
	'CGA':'R', 'CGC':'R', 'CGG':'R', 'CGT':'R',
	'GTA':'V', 'GTC':'V', 'GTG':'V', 'GTT':'V',
	'GCA':'A', 'GCC':'A', 'GCG':'A', 'GCT':'A',
	'GAC':'D', 'GAT':'D', 'GAA':'E', 'GAG':'E',
	'GGA':'G', 'GGC':'G', 'GGG':'G', 'GGT':'G',
	'TCA':'S', 'TCC':'S', 'TCG':'S', 'TCT':'S',
	'TTC':'F', 'TTT':'F', 'TTA':'L', 'TTG':'L',
	'TAC':'Y', 'TAT':'Y', 'TAA':'_', 'TAG':'_',
	'TGC':'C', 'TGT':'C', 'TGA':'_', 'TGG':'W',
	}



명령문	genetic_code['ATA']
결과	'I'

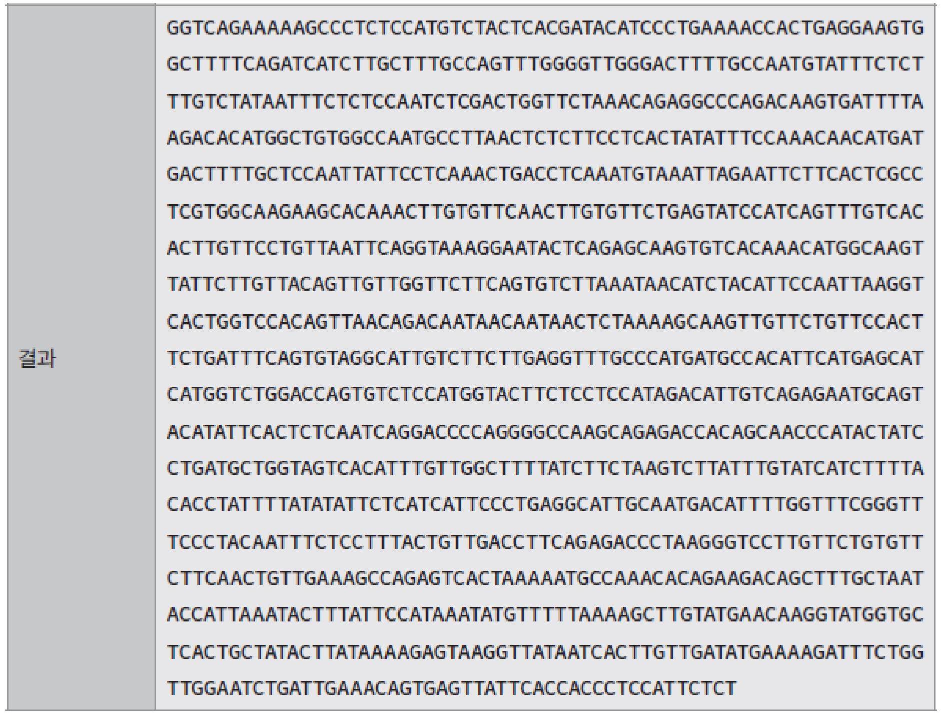
DNA를 아미노산으로 변환

- **read_seq()**
  - 다운로드한 데이터를 가공해 파일에 저장
- **NM_207618.2.fasta 파일의 첫 줄 삭제**



명령문

```
def read_seq(inputfile):
    with open(inputfile, 'r') as f:
        sequence = f.read()
    sequence = sequence.replace(' ', '')
    sequence = sequence.replace('\n', '')
    sequence = sequence.replace('\r', '')
    return sequence
with open('NM_207618.2.fasta', 'r') as inf:
    data = inf.read().splitlines(True)
with open('dna.txt', 'w') as outf:
    outf.writelines(data[1:])
dna = read_seq('dna.txt')
print(dna)
```

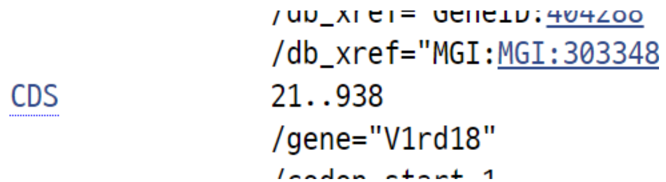


결과

```
GGTCAGAAAAAGCCCTCTCCATGTCTACTCACGATACATCCCTGAAAACCACTGAGGAAGTG
GCTTTTCAGATCATCTTGCTTTGCCAGTTTGGGGTTGGGACTTTTGCCAATGTATTTCTCT
TTGTCTATAATTTCTCTCCAATCTCGACTGGTTCTAAACAGAGGCCCAGACAAGTGATTTTA
AGACACATGGCTGTGGCCAATGCCTTAACTCTCTTCCTCACTATATTTCCAAACAACATGAT
GACTTTTGCTCCAATTATTCCTCAAACTGACCTCAAATGTAAATTAGAATTCTTCACTCGCC
TCGTGGCAAGAAGCACAAACTTGTGTTCAACTTGTGTTCTGAGTATCCATCAGTTTGTCAC
ACTTGTTCCTGTTAATTCAGGTAAAGGAATACTCAGAGCAAGTGTCACAAACATGGCAAGT
TATTCTTGTTACAGTTGTTGGTTCTTCAGTGTCTTAAATAACATCTACATTCCAATTAAGGT
CACTGGTCCACAGTTAACAGACAATAACAATAACTCTAAAAGCAAGTTGTTCTGTTCCACT
TCTGATTTCAGTGTAGGCATTGTCTTCTTGAGGTTTGCCCATGATGCCACATTCATGAGCAT
CATGGTCTGGACCAGTGTCTCCATGGTACTTCTCCTCCATAGACATTGTCAGAGAATGCAGT
ACATATTCACTCTCAATCAGGACCCCAGGGGCCAAGCAGAGACCACAGCAACCCATACTATC
CTGATGCTGGTAGTCACATTTGTTGGCTTTTATCTTCTAAGTCTTATTTGTATCATCTTTTA
CACCTATTTTATATATTCTCATCATTCCCTGAGGCATTGCAATGACATTTTGGTTTCGGGTT
TCCCTACAATTTCTCCTTTACTGTTGACCTTCAGAGACCCTAAGGGTCCTTGTTCTGTGTT
CTTCAACTGTTGAAAGCCAGAGTCACTAAAAATGCCAAACACAGAAGACAGCTTTGCTAAT
ACCATTAAATACTTTATTCCATAAATATGTTTTTAAAAGCTTGTATGAACAAGGTATGGTGC
TCACTGCTATACTTATAAAAGAGTAAGGTTATAATCACTTGTTGATATGAAAAGATTTCTGG
TTGGAATCTGATTGAAACAGTGAGTTATTCACCACCCTCCATTCTCT
```

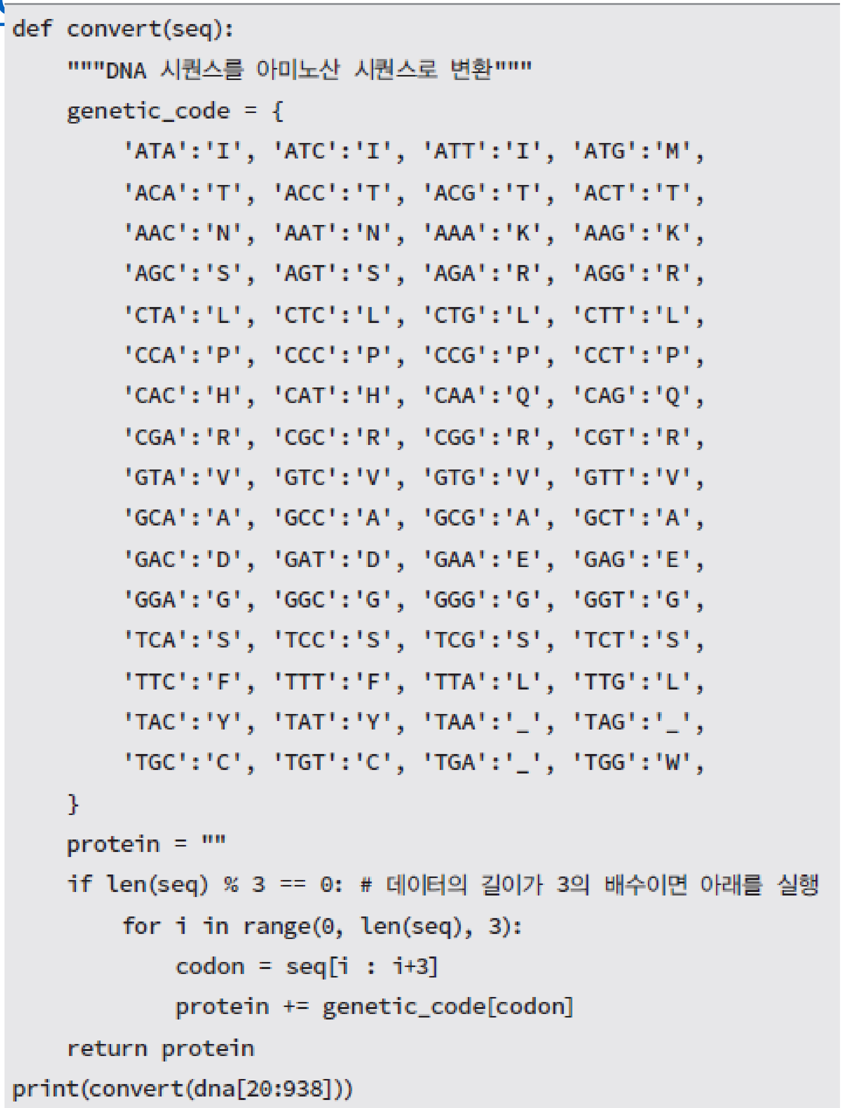
DNA를 아미노산으로 변환

- 사이트에서
  - https://www.ncbi.nlm.nih.gov/search/all/?term=NM_207618.2
  - Genomes → Nucleotide



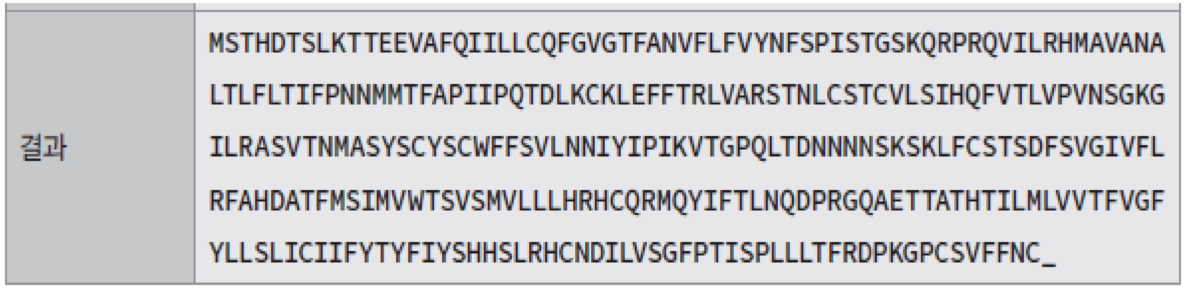
```
                     /db_xref="GeneID:404288"
                     /db_xref="MGI:303348"
     CDS             21..938
                     /gene="V1rd18"
                     /codon_start=1
```

- convert(): DNA 정보를 아미노산 시퀀스로 변환
- 아미노산의 DNA에 해당하는 부분은 [20:938]



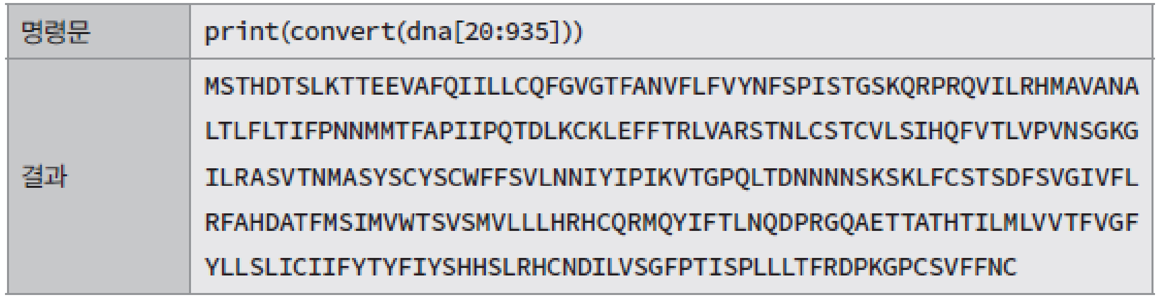
```
def convert(seq):
    """DNA 시퀀스를 아미노산 시퀀스로 변환"""
    genetic_code = {
        'ATA':'I', 'ATC':'I', 'ATT':'I', 'ATG':'M',
        'ACA':'T', 'ACC':'T', 'ACG':'T', 'ACT':'T',
        'AAC':'N', 'AAT':'N', 'AAA':'K', 'AAG':'K',
        'AGC':'S', 'AGT':'S', 'AGA':'R', 'AGG':'R',
        'CTA':'L', 'CTC':'L', 'CTG':'L', 'CTT':'L',
        'CCA':'P', 'CCC':'P', 'CCG':'P', 'CCT':'P',
        'CAC':'H', 'CAT':'H', 'CAA':'Q', 'CAG':'Q',
        'CGA':'R', 'CGC':'R', 'CGG':'R', 'CGT':'R',
        'GTA':'V', 'GTC':'V', 'GTG':'V', 'GTT':'V',
        'GCA':'A', 'GCC':'A', 'GCG':'A', 'GCT':'A',
        'GAC':'D', 'GAT':'D', 'GAA':'E', 'GAG':'E',
        'GGA':'G', 'GGC':'G', 'GGG':'G', 'GGT':'G',
        'TCA':'S', 'TCC':'S', 'TCG':'S', 'TCT':'S',
        'TTC':'F', 'TTT':'F', 'TTA':'L', 'TTG':'L',
        'TAC':'Y', 'TAT':'Y', 'TAA':'_', 'TAG':'_',
        'TGC':'C', 'TGT':'C', 'TGA':'_', 'TGG':'W',
    }
    protein = ""
    if len(seq) % 3 == 0: # 데이터의 길이가 3의 배수이면 아래를 실행
        for i in range(0, len(seq), 3):
            codon = seq[i : i+3]
            protein += genetic_code[codon]
    return protein
print(convert(dna[20:938]))
```

DNA를 아미노산으로 변환



결과	MSTHDTSLKTTEEVAFQIILLCQFGVGTFANVFLFVYNFSPISTGSKQRPRQVILRHMAVANA LTLFLTIFPNNMMTFAPIIPQTDLKCKLEFFTRLVARSTNLCSTCVLSIHQFVTLVPVNSGKG ILRASVTNMASYSCYSCWFFSVLNNIYIPIKVTGPQLTDNNNNSKSKLFCSTSDFSVGIVFL RFAHDATFMSIMVWTSVSMVLLLHRHCQRMQYIFTLNQDPRGQAETTATHTILMLVVTFVGF YLLSLICIIFYTYFIYSHHSLRHCNDILVSGFPTISPLLLTFRDPKGPCSVFFNC_

- 마지막 3개의 종결문자 제외

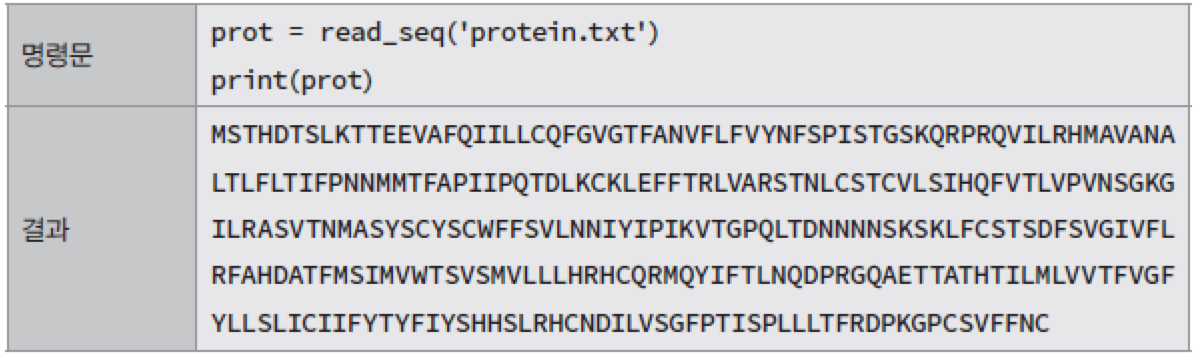


명령문	print(convert(dna[20:935]))
결과	MSTHDTSLKTTEEVAFQIILLCQFGVGTFANVFLFVYNFSPISTGSKQRPRQVILRHMAVANA LTLFLTIFPNNMMTFAPIIPQTDLKCKLEFFTRLVARSTNLCSTCVLSIHQFVTLVPVNSGKG ILRASVTNMASYSCYSCWFFSVLNNIYIPIKVTGPQLTDNNNNSKSKLFCSTSDFSVGIVFL RFAHDATFMSIMVWTSVSMVLLLHRHCQRMQYIFTLNQDPRGQAETTATHTILMLVVTFVGF YLLSLICIIFYTYFIYSHHSLRHCNDILVSGFPTISPLLLTFRDPKGPCSVFFNC

변환 결과 확인

- **변환한 아미노산 서열과 사이트에서 다운로드한 것이 일치하는지를 비교**

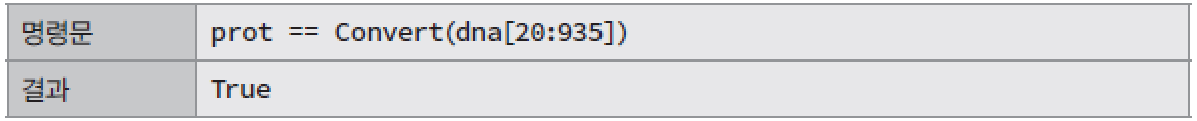
  - protein.txt



명령문	`prot = read_seq('protein.txt')`
	`print(prot)`

결과	MSTHDTSLKTTEEVAFQIILLCQFGVGTFANVFLFVYNFSPISTGSKQRPRQVILRHMAVANA LTLFLTIFPNNMMTFAPIIPQTDLKCKLEFFTRLVARSTNLCSTCVLSIHQFVTLVPVNSGKG ILRASVTNMASYSCYSCWFFSVLNNIYIPIKVTGPQLTDNNNNSKSKLFCSTSDFSVGIVFL RFAHDATFMSIMVWTSVSMVLLLHRHCQRMQYIFTLNQDPRGQAETTATHTILMLVVTFVGF YLLSLICIIFYTYFIYSHHSLRHCNDILVSGFPTISPLLLTFRDPKGPCSVFFNC

  - 직접 변환한 내용과 비교



명령문	`prot == Convert(dna[20:935])`
결과	True