목차

생물 정보학 프로그래밍

1일 차: 표준 입출력과 조건문 사용에 대한 프로그래밍 연습 I

1. "This is sequence file"을 다음과 같이 출력한다.

```
$ python bioinformatics_1_1.py
This is sequence file
$
```

2. 문자열 입력받고 받은 문자열을 다음과 같이 출력하는 .

```
$ python bioinformatics_1_2.py
Enter a string: Good day, everyone!
Good day, everyone!
$
```

3. 두 정수를 입력받아 합을 구하여 다음과 같이 출력한다.

```
$ python bioinformatics_1_3.py
Enter a integer: 10
Enter another: 9
Sum: 19
$
```

4. 두 정수를 입력받아 큰 수를 출력하는 프로그램을 작성하시오.

```
$ python bioinformatics_1_3.py
Enter a integer: 10
Enter another: 9
Sum: 19
$
```

2일 차: 표준 입출력과 조건문 사용에 대한 프로그래밍 연습 II

1. 구구단의 단 수를 입력받아 출력하는 프로그램을 작성하시오. 단, 단 수는 1과 9사이의 수만을 입력받아야 한다.

```
$ python bioinformatics_2_1.py
Which times table: 5
5 * 1 =  5
5 * 2 = 10
5 * 3 = 15
5 * 4 = 20
5 * 5 = 25
5 * 6 = 30
5 * 7 = 35
5 * 8 = 40
5 * 9 = 45
$
$ python bioinformatics_2_1.py
Which times table: 12
What?
$
```

2. 문자열을 입력받아 포함된 문자의 갯수를 출력하는 프로그램을 완성하시오.

```
$ python bioinformatics_2_2.py
Enter a string: Good day, everyone!
The string length is 19.
$
```

3. 문자열과 숫자를 입력받아 입력받은 문자열을 입력한 수만큼 반복하여 출력하는 프로그램을 완성하시오.

```
$ python bioinformatics_2_3.py
Enter a string: I am a biologist.
How many times to repeat: 3
I am a biologist.I am a biologist.I am a biologist.
$
```

4. 두 개의 문자열 입력받아 서로 다르면 두 문자열을 연결하여 출력하고, 같으면 "Two strings are identical."를 출력

하는 프로그램을 완성하시오.

```
$ python bioinformatics_2_4.py
Enter a string: I am a computer scientist.
Enter another: I am a biologist.
I am a computer scientist.I am a biologist.
$
$ python bioinformatics_2_4.py
Enter a string: I am a bioinformatician.
Enter another: I am a bioinformatician.
Two strings are identical.
$
```

5. 두 개의 문자열 `s1` 과 `s2` 를 입력받아 `s1` 의 길이가 홀수이고 `s2` 보다 짧으면 `s1` , `s2` 의 순서로 출력하고, 그렇지 않으면 반대 순서로 출력하는 프로그램을 완성하시오.

```
$ python bioinformatics_2_5.py
Enter s1: 12345
Enter s2: ABCDEF
12345ABCDEF
$
$ python bioinformatics_2_5.py
Enter s1: 12345
Enter s2: ABCDE
ABCDE12345
$
$ python bioinformatics_2_5.py
Enter s1: 123456
Enter s2: ABCDE
ABCDE123456
$
```

6. 문자열을 입력받아 역순으로 출력하는 프로그램을 작성하시오.

```
$ python bioinformatics_2_6.py
Enter a string: I am a biologist.
Reversed string: .tsigoloib a ma I
$
```

7. 다음과 같이 3-base codon과 아미노산을 입력받아 테이블을 만든다. 단, codon에 `xxx` 를 입력하면 테이블 만들기를 종료한다. (입력에 필요한 codon과 이에 해당되는 아미노산은 다음의 테이블을 참조한다.) 이제 codon을 입력받으면 테이블에서 입력한 codon에 해당되는 아미노산을 출력한다. 단, 3-base codon에 `xxx` 를 입력하면 종료한다.

```
$ python bioinformatics_2_7.py
Enter three-base codon to build: GCT
Enter amino acid: A
Enter three-base codon to build: GCC
Enter amino acid: A
...
Enter three-base codon to build: TAG
Enter amino acid: *
Enter three-base codon to build: XXX
Okay, I will switch.
Enter three-base codon to search: CCG
Amino acid for CCG: P
Enter three-base codon to search: GAC
Amino acid for GAC: D
Enter three-base codon to search: XXX
Okay, I will stop.
$
```

Amino acid	**codons**	**Amino acid**	**codons**
Ala/A	`GCT`, `GCC`, `GCA`, `GCG`	Leu/L	`TTA`, `TTG`, `CTT`, `CTC`, `CTA`, `CTG`
Arg/R	`CGT`, `CGC`, `CGA`, `CGG`, `AGA`, `AGG`	Lys/K	`AAA`, `AAG`
Asn/N	`AAT`, `AAC`	Met/M	`ATG`
Asp/D	`GAT`, `GAC`	Phe/F	`TTT`, `TTC`
Cys/C	`TGT`, `TGC`	Pro/P	`CCT`, `CCC`, `CCA`, `CCG`
Gln/Q	`CAA`, `CAG`	Ser/S	`TCT`, `TCC`, `TCA`, `TCG`, `AGT`, `AGC`
Glu/E	`GAA`, `GAG`	Thr/T	`ACT`, `ACC`, `ACA`, `ACG`
Gly/G	`GGT`, `GGC`, `GGA`, `GGG`	Trp/W	`TGG`
His/H	`CAT`, `CAC`	Tyr/Y	`TAT`, `TAC`
Ile/I	`ATT`, `ATC`, `ATA`	Val/V	`GTT`, `GTC`, `GTA`, `GTG`
Met/M	`ATG`	Stop/*	`TAA`, `TGA`, `TAG`

3일 차: 파일 핸들 및 Fasta 포멧 파일 파싱

1. "This is a sequence file."이라는 내용을 가진 `title.txt` 파일을 만든다.

```
$ python bioinformatics_3_1.py
$ cat title.txt
This is a sequence file.
$
```

2. 위에서 만든 `title.txt` 파일의 첫째줄을 읽어 다음과 같이 출력한다.

```
$ python bioinformatics_3_2.py
The first line is: This is a sequence file.
$
```

3. https://www.ncbi.nlm.nih.gov/protein/AAB07223.1?report=fasta 에서 protein FASTA 파일을 받아 `sequence.protein.fasta` 란 이름으로 저장한 후 그 내용을 읽어 다음과 같이 출력한다.

```
$ python bioinformatics_3_3.py
>AAB07223.1 BRCA2 [Homo sapiens]
MPIGSKERPTFFEIFKTRCNKADLGPISLNWFEELSSEAPPYNSEPAEESEHKNNNYEPNLFKTPQRKPS
...
QFISVSESTRTAPTSSEDYLRLKRRCTTSLIKEQESSQASTEECEKNKQDTITTKKYI
$
```

4. 위의 `sequence.protein.fasta` 파일의 내용을 `sequence.protein.2.fasta` 파일로 출력한다.

```
$ python bioinformatics_3_4.py
$ cat sequence.protein.2.fasta
>AAB07223.1 BRCA2 [Homo sapiens]
MPIGSKERPTFFEIFKTRCNKADLGPISLNWFEELSSEAPPYNSEPAEESEHKNNNYEPNLFKTPQRKPS
...
QFISVSESTRTAPTSSEDYLRLKRRCTTSLIKEQESSQASTEECEKNKQDTITTKKYI
$
```

5. 위에서 만든 `sequence.protein.2.fasta` 의 두번째줄을 다음과 같이 출력한다.

```
$ python bioinformatics_3_5.py
The second line is: MPIGSKERPTFFEIFKTRCNKADLGPISLNWFEELSSEAPPYNSEPAEESEHKNN
$
```

6. `sequence.protein.2.fasta` 의 첫째줄은 `title` 이란 변수에, 둘째줄부터 끝까지는 개행문자를 없앤

후 `seq` 이란 변수에 저장하고 다음과 같이 출력한다.

```
$ python bioinformatics_3_6.py
title: >AAB07223.1 BRCA2 [Homo sapiens]
seq: MPIGSKERPT...QDTITTKKYI
$
```

7. 위에서 만든 `seq` 문자열의 위치를 물어보고, 입력된 위치에서 시작하는 3개의 아미노산을 다음과 같이 출력한다. `XXX` 를 입력할 때 까지 이 작업을 반복한다.

```
$ python bioinformatics_3_7.py
Position: 15
Three amino acids: FKT
Position: 20
Three amino acids: NKA
Position: XXX
Okay, I will stop.
$
```

8. 찾고자 하는 아미노산을 물어보고, 입력된 아미노산을 가진 `seq` 문자열의 모든 위치를 다음과 같이 출력한다. `XXX` 를 입력할 때 까지 이 작업을 반복한다.

```
$ python bioinformatics_3_8.py
Searching for: S
Found at: 5, 28, 36, 37, 44, ..., 3376, 3389, 3396, 3397, 3400
Searching for: Q
Found at: 66, 73, 84, 92, 126, ..., 3350, 3361, 3394, 3398, 3409
Searching for: XXX
Okay, I will stop.
$
```

4일차: GenBank 포맷 파싱, 문자열 포맷팅, 서열 핸들(상보적서열, 역서열, 등 생성), 폴드 핸들

1. `https://www.ncbi.nlm.nih.gov/protein/AAB07223.1?report=genpept` 에서 protein GenBank 파일을 받아 `sequence.protein.gb` 란 이름으로 저장한 후 첫째줄은 `title` 이란 변수에, 'ORIGIN'이라는 단어로 시작되는 행을 찾아, 그 다음 행부터 끝까지 읽어 `seq` 이라는 변수에 저장하고 다음과 같이 출력한다.

```
$ python bioinformatics_4_1.py
title: LOCUS       AAB07223                3418 aa            linear   PRI
seq:         1 mpigskerpt ffeifktrcn kadlgpisln wfeelsseap pynsepaees ehknn
...
   3361 qfisvsestr taptssedyl rlkrrcttsl ikeqessqas teeceknkqd tittkkyi
//
$
```

2. 위에서 만든 `seq` 문자열에서 알파벳만 저장하여 다음과 같이 출력한다.

```
$ python bioinformatics_4_2.py
title: LOCUS       AAB07223                3418 aa            linear   PRI
seq: mpigskerptffeifktrcnkadlgpisln...ikeqessqasteeceknkqdtittkkyi
$
```

3. 위에서 만든 `seq` 문자열을 한 줄에 70개의 문자를 포맷팅하여 다음과 같이 출력한다.

```
$ python bioinformatics_4_3.py
mpigskerptffeifktrcnkadlgpislnwfeelsseappynsepaeesehknnnyepnlfktpqrkps
...
qfisvsestrtaptssedylrlkrrcttslikeqessqasteeceknkqdtittkkyi
$
```

4. `https://www.ncbi.nlm.nih.gov/nuccore/NM_001276699.1?report=fasta` 에서 nucleotide FASTA 파일을 받아 `sequence.nucleotide.fasta` 란 이름으로 저장하고, `https://www.ncbi.nlm.nih.gov/nuccore/NM_001276699.1?report=genbank` 에서 nucleotide GenBank 파일을 받아 `sequence.nucleotide.gb` 란 이름으로 저장한 후 입력파일명, 출력파일명, 옵션을 입력받아 아래와 같이 동작하는 프로그램을 작성하시오. (option-3의 경우 FASTA format의 header line은 GeneBank format의 첫 번째 줄로 한다.)

```
$ python bioinformatics_4_4.py
Enter input file: sequence.nucleotide.fasta
Enter ouput file: reverse.sequence.nucleotide.fasta
Option-1) Read a FASTA format DNA sequence file and make a reverse sequence
Option-2) Read a FASTA format DNA sequence file and make a reverse compleme
Option-3) Convert GenBank format file to FASTA format file.
Select the option: 1
$ cat reverse.sequence.nucleotide.fasta
>NM_001276699.1 Homo sapiens tumor protein p53 (TP53), transcript variant 7
GTGGGGAGTCTGTGTGTCCACCGTCGTTTCAAAATAACATTTTATTCTCTAGCTATATTTTTACCCTATA
...
TCGGAGGTAGAGGACCGGAGT
$
$ python bioinformatics_4_4.py
Enter input file: sequence.nucleotide.fasta
Enter ouput file: reverse.complement.sequence.nucleotide.fasta
Option-1) Read a FASTA format DNA sequence file and make a reverse sequence
Option-2) Read a FASTA format DNA sequence file and make a reverse compleme
Option-3) Convert GenBank format file to FASTA format file.
Select the option: 2
$ cat reverse.sequence.nucleotide.fasta
>NM_001276699.1 Homo sapiens tumor protein p53 (TP53), transcript variant 7
CACCCCTCAGACACACAGGTGGCAGCAAAGTTTTATTGTAAAATAAGAGATCGATATAAAAATGGGATAT
...
AGCCTCCATCTCCTGGCCTCA
$
$ python bioinformatics_4_4.py
Enter input file: sequence.nucleotide.gb
Enter ouput file: new.sequence.nucleotide.fasta
Option-1) Read a FASTA format DNA sequence file and make a reverse sequence
Option-2) Read a FASTA format DNA sequence file and make a reverse compleme
Option-3) Convert GenBank format file to FASTA format file.
Select the option: 3
$ cat new.sequence.nucleotide.fasta
>LOCUS       NM_001276699            2331 bp    mRNA    linear   PRI 28-DEC
tgaggccaggagatggaggctgcagtgagctgtgatcacaccactgtgctccagcctgagtgacagagca
...
cacctgtgtgtctgaggggtg
$
```

5. 위의 `sequence.nucleotide.gb` 파일의 내용 중 'Title'에 해당하는 문자열을 읽어 들인 후, 읽은 순서대로 각 내용당 한 줄로 만들어 다음과 같이 출력한다.

```
$ python bioinformatics_4_5.py
  TITLE     G-quadruplex structures in TP53 intron 3: role in alternative s
            Delta160p53 is a novel N-terminal p53 isoform encoded by Delta1
            ...
            Isolation and characterization of a human p53 cDNA clone: expre
$
```

5일차: 6-frame translation

1. 위의 `sequence.nucleotide.fasta` 파일의 sequence를 한 줄에 3-base codon 단위로 다음과 같이 출력한다. (끝에 codon 이 만들어지지 않는 부분은 출력하지 않는다.)

```
$ python bioinformatics_5_1.py
TGA
GGC
CAG
...
TGA
GGG
GTG
$
```

2. 위의 `sequence.nucleotide.fasta` 파일의 sequence를 한 줄에 60 bases 단위로 출력하고 각 줄당 해당하는 아미노산으로 translation 하여 다음과 같이 출력한다.

```
$ python bioinformatics_5_2.py
TGAGGCCAGGAGATGGAGGCTGCAGTGAGCTGTGATCACACCACTGTGCTCCAGCCTGAG
*  G  Q  E  M  E  A  A  V  S  C  D  H  T  T  V  L  Q  P  E
TGACAGAGCAAGACCCTATCTCAAAAAAAAAAAAAAAAAAAGAAAAGCTCCTGAGGTGTAG
*  Q  S  K  T  L  S  Q  K  K  K  K  K  R  K  A  P  E  V  *
...
ATCTCTTATTTTACAATAAAACTTTGCTGCCACCTGTGTGTCTGAGGGGTG
I  S  Y  F  T  I  K  L  C  C  H  L  C  V  *  G  V
$
```

3. 위의 `sequence.nucleotide.fasta` 파일의 sequence를 이용하여 translation을 6개의 reading frame으로 다음과 같이 출력한다. (출력순서 : forward 1, forward 2, forward 3, sequence, reverse complement sequence, reverse 1, reverse 2, reverse 3)

```
$ python bioinformatics_5_3.py
*  G  Q  E  M  E  A  A  V  S  ...L  C  C  H  L  C  V  *  G  V
 E  A  R  R  W  R  L  Q  *  A ... F  A  A  T  C  V  S  E  G
  R  P  G  D  G  G  C  S  E  L...  L  L  P  P  V  C  L  R  G
TGAGGCCAGGAGATGGAGGCTGCAGTGAGC...CTTTGCTGCCACCTGTGTGTCTGAGGGGTG
ACTCCGGTCCTCTACCTCCGACGTCACTCG...GAAACGACGGTGGACACACAGACTCCCCAC
  S  A  L  L  H  L  S  C  H  A...  K  A  A  V  Q  T  D  S  P  H
    P  W  s  I  s  A  A  T  L ... S  Q  Q  W  R  H  T  Q  P  T
   L  G  P  S  P  P  Q  L  S  ...V  K  S  G  G  T  H  R  L  P
$
```

4. 위의 `sequence.nucleotide.gb` 파일을 읽어 CDS 영역의 정보를 찾아 sequence에서 해당 서열을 translation 한 결과를 다음과 같이 출력한다.

```
$ python bioinformatics_5_4.py
MAIYKQSQHMTEVVRRCPHHERCSDSDGLAPPQHLIRVEGNLRVEYLDDRNTFRHSVVVPYEPPEVGSDC
TTIHYNYMCNSSCMGGMNRRPILTIITLEDSSGNLLGRNSFEVRVCACPGRDRRTEEENLRKKGEPHHEL
PPGSTKRALPNNTSSSPQPKKKPLDGEYFTLQMLLDLRWCYFLINSS*
$
```