|  |
| --- |
| C++프로그래밍및실습  DNA 해독기  **진척 보고서 #1** |

|  |  |
| --- | --- |
|  | 제출일자: 2023.11.25  제출자명: 백성준  제출자학번: 175989 |

1. 프로젝트 목표
   1. 배경 및 필요성

“Chat gpt"와 같은 대화형 인공지능의 출현으로 세계의 기술적 환경은 크게 변화하였다. 평범한 사람들도 이제는 그 변화를 피부로 느끼고 있다. 당장 이러한 대화형 인공지능을 사용해본 사람들은 그 용이성과 위기감을 느낄 수 있다. 머지않아 대부분의 단순 노동은 대체될 것이며, 가장 인간적이라 생각한 부분도 대체될 것이다. 그리고 실제로 지금도 대체가 일어나고 있다.

이러한 변화 속에서 인공지능 또는 자동화 기술을 다양한 학문과 연계한다면 큰 경쟁력을 획득할 수 있을 것이다. 본 프로젝트에서는 생물학에서의 자동화를 목표로 한다.

* 1. 프로젝트 목표

이 프로젝트에서는 생명의 가장 기초적이지만 근본적인 DNA 염기서열을 자동으로 처리하는 프로그램을 다루고자 한다. DNA 염기서열에서 ORF(Open Reading Frame)을 찾아내고 CDS(Coding sequence)를 최종적으로 얻어내는 것을 목표로 한다. 또한, 이 CDS를 단백질로 변환하는 작업까지 수행한다. 이 때, 사람의 DNA라 가정하며 가장 자주 발견되는 consensus sequence를 따른다고 가정한다. 또한, codon table은 보편적인 table을 사용한다.

* 1. 차별점

기존의 염기서열 정보를 단백질 서열로 변환하는 프로그램에는 단순 변환만 존재하였다. 여기에서 더 나아가 DNA에서 얻어낸 단백질의 정보가 저장된 ORF(Open Reading Frame)에서 intron을 제거해낼 수 있다. 또한, 얻어낸 서열 정보의 분석을 진행한다.

* 1. 배경지식

사람의 유전자는 A,T,G,C의 염기를 가지고 있다. 이 염기가 3 개 모여서 단백질 정보를 이루는 것을 codon이라 하고, codon은 20 개 단백질 정보를 담고 있다. 각 codon에 하나의 단백질이 상응하며 단백질의 시작이 되는 start codon과 stop codon이 각각 존재한다. Start codon에서 stop codon 까지를 ORF(Open Reading Frame)라 한다.

DNA는 RNA로 변환되어 splicing이라는 intron이 제거되는 과정을 거친다. Intron이 제거된 RNA가 ribosome에서 단백질로 변환된다. 이때의 RNA서열에 상응하는 DNA를 CDS(Coding Sequence)라 한다.

1. 기능 계획
2. 사용자의 이름과 염기서열 정보 저장.

사용자의 이름을 입력할 수 있다.

염기서열의 정보를 입력할 수 있다.

1. 염기서열의 정보를 분석
2. 염기서열에서 ORF를 저장

-6 가지의 가능한 reading frame을 저장한다.

-각 서열에서 ATG로 시작하고, TGA, TAA 또는 TAG로 끝나는 서열을 저장한다.

1. 얻어낸 ORF에서 intron을 제거

- [GTA][AG ] 또는 [GTG][AG ]로 시작하며 [CAG]로 끝나는 서열을 제거한다. 이 때, 중간에 A가 하나 이상 존재해야 한다.

1. Intron이 제거된 CDS를 단백질 형태의 정보로 변환

-codon table에 맞게 염기서열을 단백질 형태로 변환한다. 즉, 각 codon에 맞는 amino acid로 변환한다.

1. Kozak sequence score를 계산

-염기서열에서 +4 위치에서 C 또는 G가 나오면 점수를 1 올리고, -2 위치에 A 또는 G가 존재하면 마찬가지로 1점 올리고, -3 에서 A 또는 G가 존재하면 +3점 증가한다. 이 점수를 합산하여 Kozak strength를 측정하여 출력시킨다.

**3. 진척사항**

**1) 기능 구현**

**(1) 사용자의 이름과 염기서열 정보, 찾을 ORF 범위 저장**

- 입출력

입력

User user: 사용자의 이름, DNA, 찾을 ORF의 크기범위 저장 클래스

string name : 사용자의 이름을 받을 변수

string sequence: 사용자의 DNA서열을 받을 변수

void User::SetName(string name):클래스 내의 사용자의 이름을 초기화하는 클래스 함수

void User::SetSequence(string sequence): 클래스 내의 사용자의 함수를 초기화하는 클래스 함수

int User::length1 : 찾을 ORF 크기의 최소값

int User::length2 : 찾을 ORF 크기의 최대값

void SetName(string name):유저의 이름을 반환하는 클래스 함수

void SetSequence(string sequence):유저의 서열을 반환하는 클래스 함수

출력

확인을 위해 클래스 내에 존재하는 서열과 이름을 출력.

- 설명

사용자의 이름을 입력 받고, DNA 서열, 찾을 ORF의 범위를 입력 받는다.

입력 받은 내용을 User클래스 user객체에 저장한다.

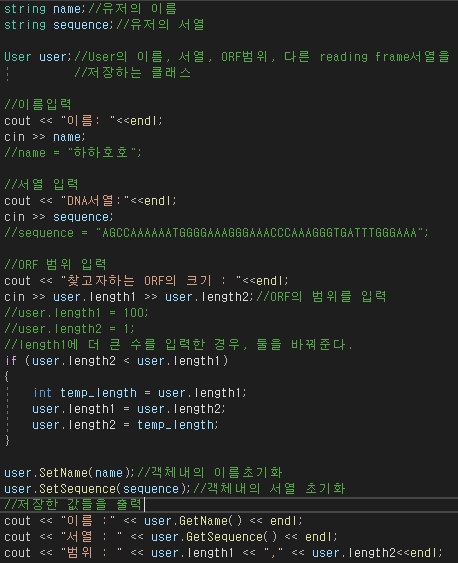
범위를 저장할 때, length1에 length2보다 더 작은 값이 저장되도록 한다.

확인을 위해 서열과 이름을 출력.

- 적용된 배운 내용

클래스, 함수, cin,cout, get/set메소드를 통한 private 변수 접근, public 함수와 변수에 “ . ”을 이용한 접근, if 조건문

- 코드 스크린샷



**(2-1)** 6 가지의 가능한 reading frame을 저장한다.

- 입출력

Void User::FrameSetting():User내의 user\_dna\_sequence에 존재하는 DNA서열을 6가지의 reading frame으로 바꿔서 클래스내에 저장하는 함수.

- 설명

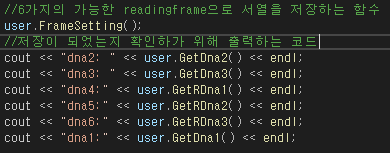
User 클래스의 멤버함수를 사용하여 User 클래스의 멤버변수에 저장.

저장된 6가지 reading frame을 가진 서열들을 출력.

- 적용된 배운 내용 (예: 반복문, 조건문, 클래스, 함수, 포인터 등)

클래스, “.”로 객체의 public 함수에 접근,

- 코드 스크린샷



**(2-2)** 각 서열에서 ATG로 시작하고, TGA, TAA 또는 TAG로 끝나는 서열을 저장한다.(ORF 저장)

- 입출력

입력

Orf user\_seq1~6 : 6가지의 reading frame의 서열 중 하나를 받아서 저장하고 이를 분석하는 객체. user\_seq1부터 6까지 6가지의 객체를 사용.

string User::GetDna1~6() : User 클래스 내의 6가지 reading frame을 반환하는 함수. GetDna1부터 GetDna6까지 6가지의 함수 존재.

vector<string> Orf::orf1 : 받은 6가지 서열을 스트링 벡터화한 것을 저장하기 위한 벡터. 3개씩 묶어서 하나의 스트링에 저장하고, 이를 이 클래스 변수에 저장.

void Orf::TransferSeq() : 받은 서열을 세 nucleotide당 하나의 스트링 객체에 저장하여 Orf클래스 내의 스트링 벡터에 저장하는 함수

void Orf::IndexFinder() : 클래스 내의 스트링 벡터에 저장된 서열에서 start codon과 stop codon에 해당하는 서열을 가진 인덱스의 값을 저장하는 함수.

void Orf::OrfFinder(User user) : 위의 함수로 얻어진 인덱스를 바탕으로 스트링 벡터에 있는 start ccodon과 stop codon이 들어있는 스트링 벡터의 인덱스를 저장한는 함수.

vector.empty() : 벡터가 비어있다면 true를 반환하는 벡터의 기본 멤버 함수.

Vector.size() : 벡터의 크기만큼 정수를 반환하는 함수.

출력

user\_seq1에서 6까지의 스트링 벡터화된 서열을 출력.

스트링화된 벡터에서 ORF를 찾아서 출력.

- 설명

각 reading frame을 조정한 서열들을 Orf 클래스에 넣어서 서열을 가공한다.

스트링 벡터를 사용하여 서열을 먼저 저장한다. 저장한 스트링 벡터를 출력.

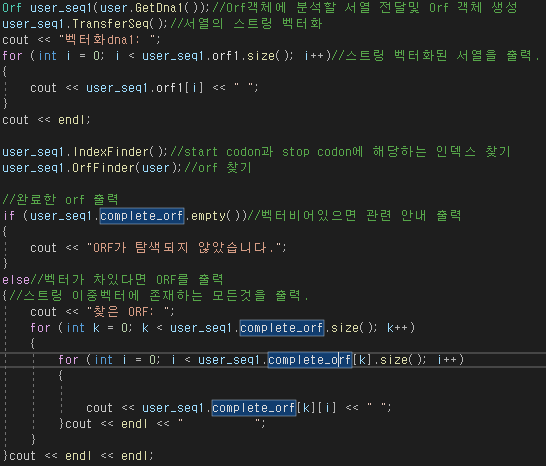
저장한 스트링 벡터에서 ORF의 시작과 끝이되는 start codon(ATG)과 stop codon(TGA,TAA,TAG)과 일치하는 스트링의 인덱스를 저장하고 이에 맞는 인덱스를 기반으로 사용자가 입력한 찾고자하는 ORF의 크기와 일치하면 그것을 이중 벡터vector<vetor<string>>에 저장한다.

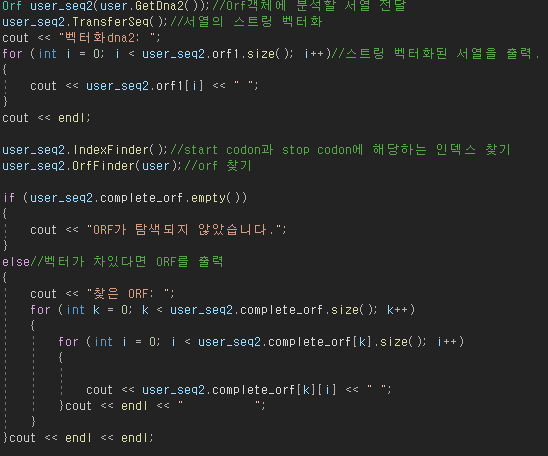
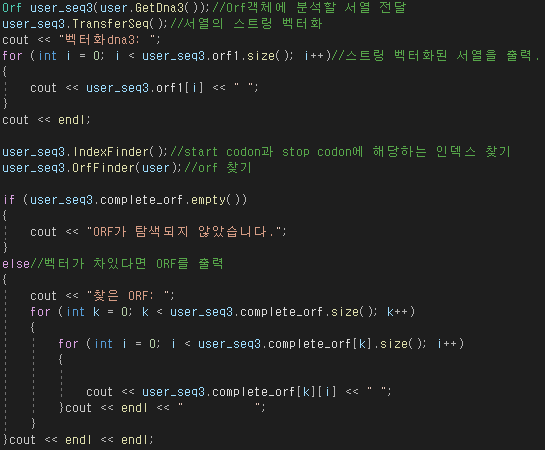
저장한 ORF를 모두 출력한다.

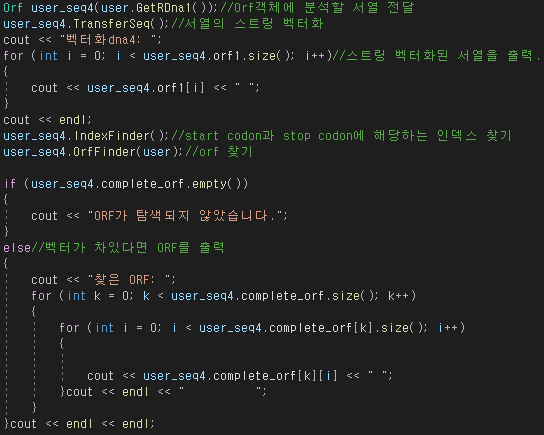
- 적용된 배운 내용 (예: 반복문, 조건문, 클래스, 함수, 포인터 등)

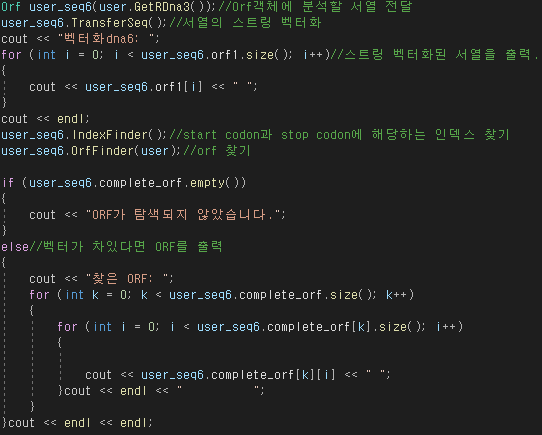
If-else 조건문, 벡터, string, 후위 연산자, 배열, for문

- 코드 스크린샷





* 1. 관련 함수 설명

1. void User::SetName(string name), void User::SetSequence(string sequence)

- 입출력

입력

String User::user\_name, string User::user\_dna\_sequence : 유저의 이름과 서열을 저장하는 변수

매개변수 string name : 객체 내의 유저 이름을 초기화하기 위한 매개변수

매개변수string sequence: 사용자의 서열을 받기 위한 매개변수

출력

없음

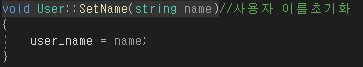
- 설명

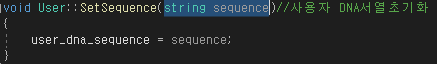
유저의 이름을 받아서 초기화 한다. 유저의 서열을 받아서 초기화 한다.

- 적용된 배운 내용 (예: 반복문, 조건문, 클래스, 함수, 포인터 등)

Get set 메소드, 클래스, string

- 코드 스크린샷





1. void User::FrameSetting()

- 입출력

입력

String temp\_dna : DNA서열을 받아 놓기 위한 변수, 원본을 훼손하지 않기 위함.

String User::dna1, String User::dna2, String User::dna3 : 유저의 정방향 reading frame 세 경우를 저장하기 위한 변수

String User:: reverse\_dna2, String User:: reverse\_dna1, String User:: reverse\_dna3 : 유저의 역방향 reading frame 세 경우를 저장하기 위한 변수

출력

없음

- 설명

유저의 정방향 서열을 따로 저장하고 앞에서 하나씩 지워서 나중에 3개씩 묶었을 때 reading frame이 바뀌도록 한다.

서열을 뒤집어서 같은 과정을 반복하여 총 6개의 서열을 객체 내의 변수에 저장한다.

- 적용된 배운 내용 (예: 반복문, 조건문, 클래스, 함수, 포인터 등)

For 문, 클래스, string, string의 erase, length 멤버 함수, += 복합 대입 연산자, --후위 감소 연산자

- 코드 스크린샷



1. string User::GetName(),string User::GetSequence(),string User::GetDna1(),string User::GetDna2(),string User::GetDna3(),string User::GetRDna1(),string User::GetRDna2(),string User::GetRDna3()

- 입출력

입력

string User::user\_name: 사용자의 이름을 저장하는 변수

string User::user\_dna\_sequence : 사용자의 서열을 저장하는 변수

string User::dna1, string User::dna2, string User::dna3, string User::reverse\_dna1, string User:: reverse\_dna2, string User:: reverse\_dna3 : 사용자의 reading frame 변경하여 저장하기 위한 변수

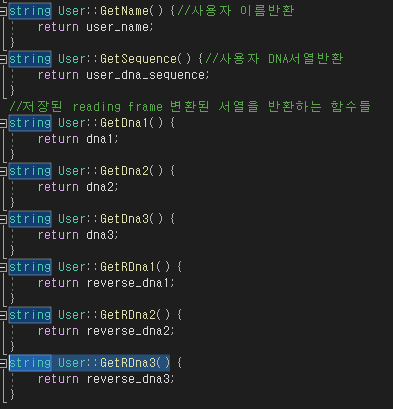
- 설명

사용자의 이름, 서열, reading frame이 바뀐 서열을 반환하는 Get 메소드.

- 적용된 배운 내용 (예: 반복문, 조건문, 클래스, 함수, 포인터 등)

Get/set 메소드, 클래스, 스트링

- 코드 스크린샷



1. void Orf::TransferSeq()

- 입출력

입력

string Orf::original\_seq : 분석할 서열을 저장해 놓기 위한 변수

string triplet : 분석할 서열에서 세 개씩 나눠서 저장할 때, 그 세 개의 서열을 담아 놓기 위한 변수

vector<string> Orf::orf1 : 세 개씩 서열을 나눠서 저장해 놓기 위한 스트링 벡터.

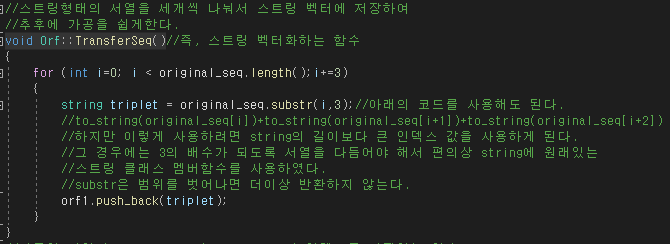
- 설명

받아온 분석할 서열을 지역변수에 저장하여 벡터에 세 개씩 나눠서 담아준다. 이러한 과정을 통해 추후 분석을 용이하게 하기 위함.

- 적용된 배운 내용 (예: 반복문, 조건문, 클래스, 함수, 포인터 등)

for문, to\_string, string, 복합 연산자, 벡터, 벡터의 멤버함수인 push\_back()

- 코드 스크린샷



1. void Orf::IndexFinder()

- 입출력

입력

vector<string> orf1 : 분석할 서열을 벡터화한 서열을 저장하고 있는 벡터. Orf 클래스에 존재

vector<int> atg\_index : start codon이 발견된 벡터의 인덱스를 저장하는 벡터. Orf 클래스에 존재

vector<int> tga\_index, vector<int> taa\_index, vector<int> tag\_index : stop codon이 발견된 벡터의 인덱스를 저장하기 위한 벡터. Orf 클래스에 존재

출력 : 없음

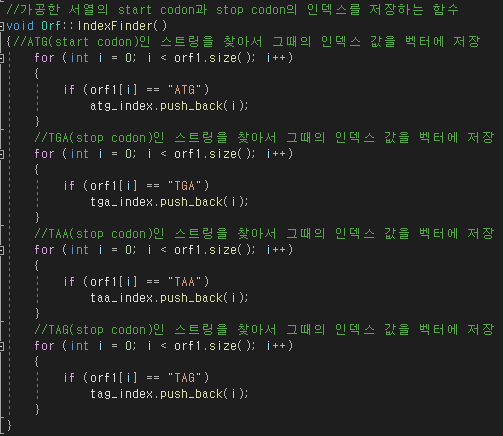
- 설명

벡터화한 분석할 서열에서 start codon과 stop codon이 있는 벡터의 string을 찾아서 각각을 대변하는 인덱스를 벡터에 저장한다.

- 적용된 배운 내용 (예: 반복문, 조건문, 클래스, 함수, 포인터 등)

배열, 벡터, 벡터의 멤버함수 push\_back(), for문 ,if 문

- 코드 스크린샷



1. void Orf::OrfFinder(User user)

- 입출력

입력

User user : 사용자가 입력한 찾을 ORF 크기의 범위를 얻어오기 위해 클래스 사용

vector<int> atg\_index : start codon이 발견된 벡터의 인덱스가 저장된 벡터. Orf 클래스에 존재

vector<int> tga\_index, vector<int> taa\_index, vector<int> tag\_index : stop codon이 발견된 벡터의 인덱스가 저장된 벡터. Orf 클래스에 존재

vetor<string> temp\_com\_orf : 필요한 부분만 저장해 놓기 위한 벡터. Orf 클래스에 존재

vector<vector<string>> complete\_orf : 찾아낸 ORF를 저장해 놓기 위한 스트링 이중 벡터.

- 설명

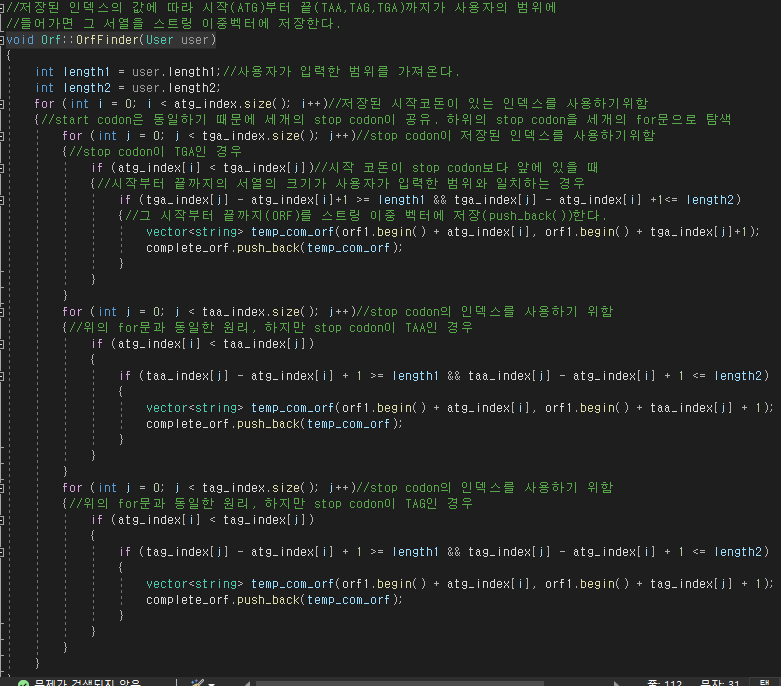
앞에서 찾은 인덱스를 바탕으로 해당하는 범위의 서열을 따로 이중 벡터에 저장한다.

사용자가 입력한 ORF크기의 범위 내에 존재하고 서열의 방향과 일치하면 그만큼의 서열을 저장한다.

- 적용된 배운 내용 (예: 반복문, 조건문, 클래스, 함수, 포인터 등)

if, for 조건문, 벡터, 스트링, 비교 연산자

- 코드 스크린샷



**2) 테스트 결과**

**(1) 사용자의 이름과 염기서열 정보, 찾을 ORF 범위 저장**

- 설명

유저의 이름과 서열, 찾고자 하는 ORF의 크기를 입력 받아서, 입력 받은 값들을 User클래스의 객체인 user에 저장한다. 이후에, 저장한 값을 출력한다.

- 테스트 결과 스크린샷

이름과 서열을 입력 받는다.



찾고자 하는 ORF 크기의 범위를 입력 받는다.



User user객체에 이름과 서열 범위를 저장한다.

입력 받은 이름과 서열을 출력한다.



입력 받은 범위를 출력한다.

Length1에 length2보다 작은 값이 저장된 경우

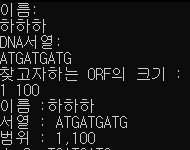


user.length1에 user.length2보다 큰 값이 저장된 경우





전체적으로 실행



**(2-1)** 6 가지의 가능한 reading frame을 저장한다.

-설명

입력 받은 서열을 6가지의 가능한 reading frame으로 변환하여 User user객체 내의 변수에 저장한다.

저장이 잘 되었는지 확인을 위해 출력한다.

-테스트결과 스크린샷

입력 받은 서열



Reading frame을 다르게 하여 저장한 6가지 서열



**(2-2)** 각 서열에서 ATG로 시작하고, TGA, TAA 또는 TAG로 끝나는 서열을 저장한다.(ORF 저장)

-설명

각 reading frame을 조정한 서열들을 Orf 클래스에 넣어서 서열을 가공한다.

스트링 벡터를 사용하여 서열을 먼저 저장한다. 저장한 스트링 벡터를 출력.

저장한 스트링 벡터에서 ORF의 시작과 끝이되는 start codon과 stop codon과 일치하는 스트링의 인덱스를 저장하고 이에 맞는 인덱스를 기반으로 사용자가 입력한 찾고자하는 ORF의 크기와 일치하면 그것을 이중 벡터vector<vetor<string>>에 저장한다.

저장한 ORF를 출력한다.

ORF를 찾지 못했으면 찾지 못했음을 알린다.

-테스트 결과 스크린샷

서열을 벡터화하고 이를 출력.



ORF가 ATG로 시작하고 TGA로 끝나는 경우



ORF가 ATG로 시작하고 TAA로 끝나는 경우



ORF가 ATG로 시작하고 TAG로 끝나는 경우



ORF를 찾지 못한 경우



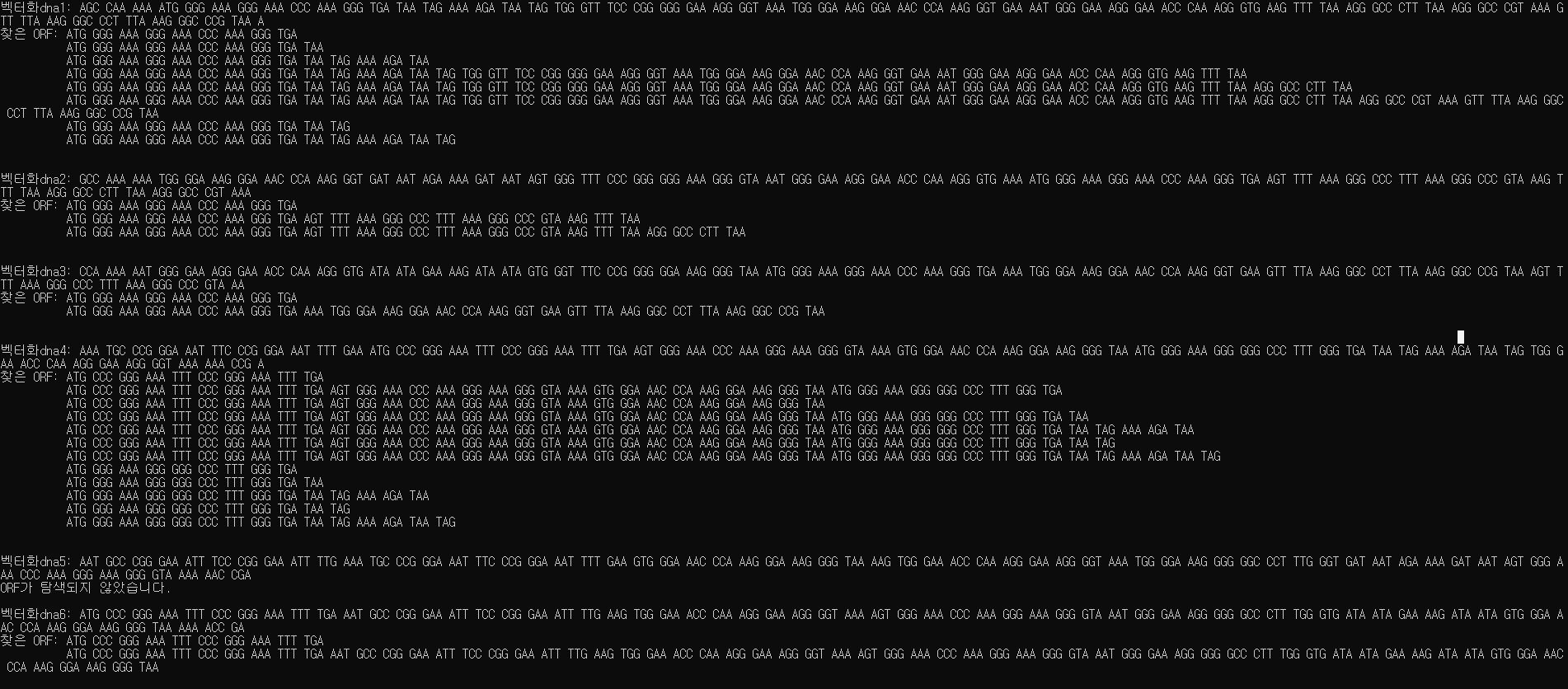
여러 ORF를 찾아낸 경우



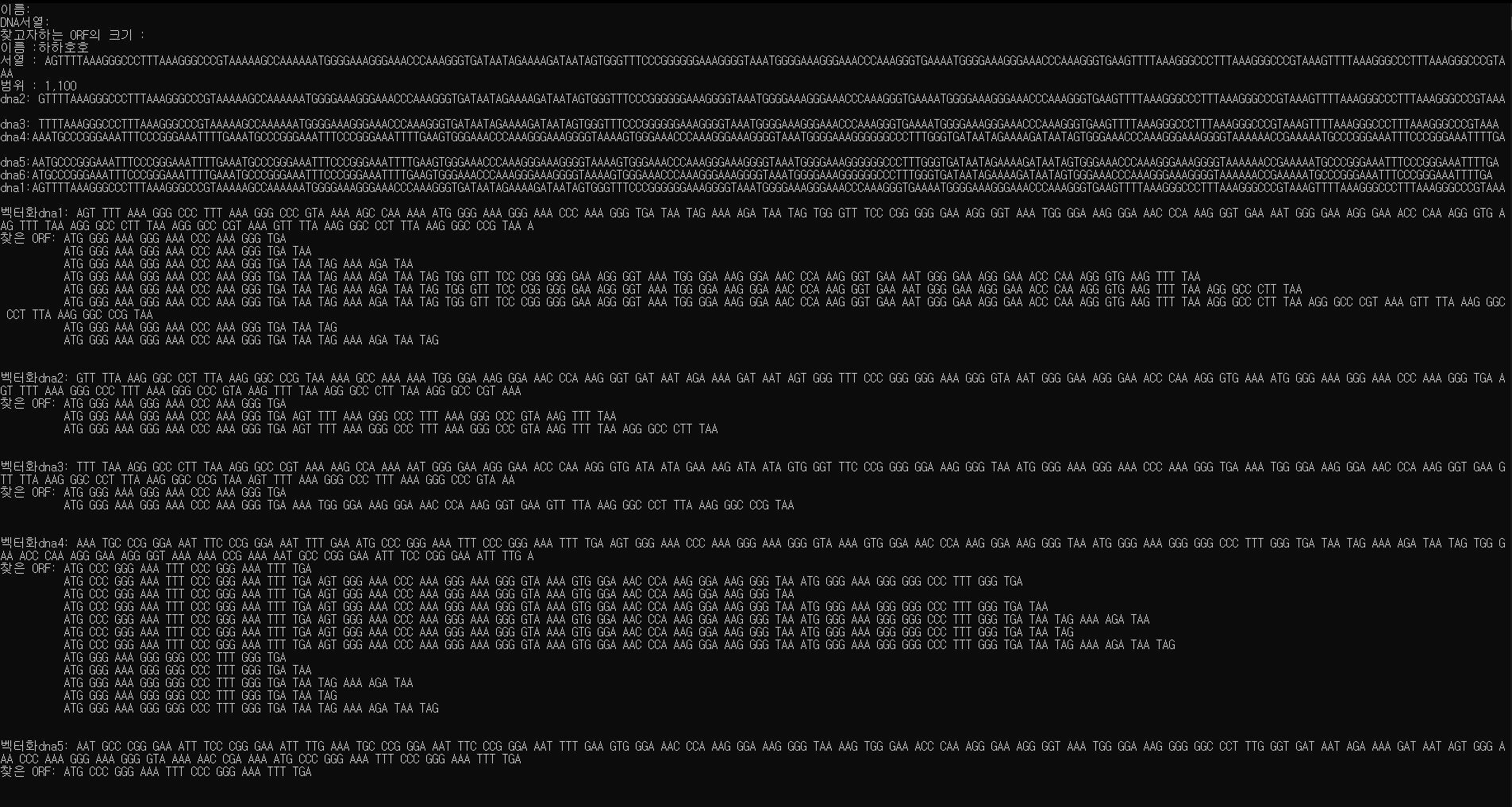
범위가 ORF보다 작은 경우

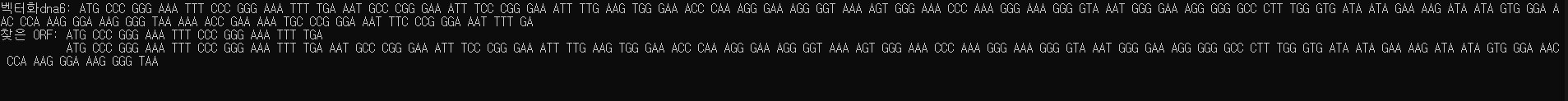


전체 스크린샷



**(2-3) 전체 스크린샷**





**4. 계획 대비 변경 사항**

1. 사용자의 이름과 염기서열 정보 저장.

- 이전

사용자의 이름을 입력할 수 있다.

염기서열의 정보를 입력할 수 있다.

- 이후

**사용자의 이름과 염기서열 정보, 찾을 ORF 범위 저장**

사용자의 이름을 입력

염기서열의 정보를 입력

찾을 ORF의 크기를 입력

- 사유

범위를 정하지 않으면 너무 많은 쓸모 없는 ORF들이 찾아지기 때문이다. 다시말해, 노이즈를 제거하는 것이다.

1. 염기서열의 정보를 분석

-이전

1. 염기서열에서 ORF를 저장

-6 가지의 가능한 reading frame을 저장한다.

-각 서열에서 ATG로 시작하고, TGA, TAA 또는 TAG로 끝나는 서열을 저장한다.

1. 얻어낸 ORF에서 intron을 제거

- [GTA][AG ] 또는 [GTG][AG ]로 시작하며 [CAG]로 끝나는 서열을 제거한다. 이 때, 중간에 A가 하나 이상 존재해야 한다.

1. Intron이 제거된 CDS를 단백질 형태의 정보로 변환

-codon table에 맞게 염기서열을 단백질 형태로 변환한다. 즉, 각 codon에 맞는 amino acid로 변환한다.

1. Kozak sequence score를 계산

-염기서열에서 +4 위치에서 C 또는 G가 나오면 점수를 1 올리고, -2 위치에 A 또는 G가 존재하면 마찬가지로 1점 올리고, -3 에서 A 또는 G가 존재하면 +3점 증가한다. 이 점수를 합산하여 Kozak strength를 측정하여 출력시킨다.

-이후

1. 6 가지의 가능한 reading frame을 저장한다.
2. 각 서열에서 ATG로 시작하고, TGA, TAA 또는 TAG로 끝나는 서열을 저장한다.(ORF 저장)
3. 얻어낸 ORF에서 intron을 제거

- [GTA][AG ] 또는 [GTG][AG ]로 시작하며 [CAG]로 끝나는 서열을 제거한다. 이 때, 중간에 A가 하나 이상 존재해야 한다.

1. Intron이 제거된 CDS를 단백질 형태의 정보로 변환

-codon table에 맞게 염기서열을 단백질 형태로 변환한다. 즉, 각 codon에 맞는 amino acid로 변환한다.

1. Kozak sequence score를 계산

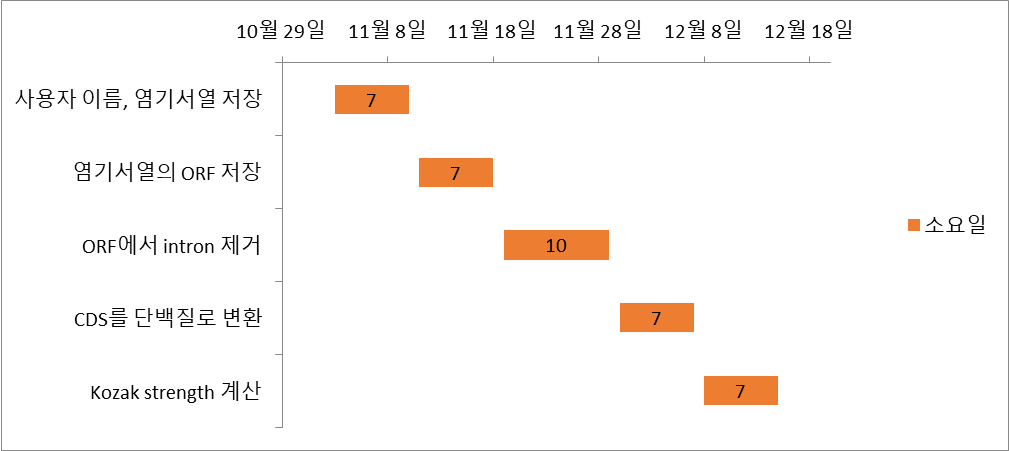
-염기서열에서 +4 위치에서 C 또는 G가 나오면 점수를 1 올리고, -2 위치에 A 또는 G가 존재하면 마찬가지로 1점 올리고, -3 에서 A 또는 G가 존재하면 +3점 증가한다. 이 점수를 합산하여 Kozak strength를 측정하여 출력시킨다.

-사유

기능을 좀 더 세분화하여 포괄적인 설명과 기능을 더 구체화하기 위함.

**5. 프로젝트 일정**

(진행한 작업과 진행 중인 작업 등을 표기)



|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 일정명 | 시작일 | 소요일 | 마감일 |  |
| 사용자 이름, 염기서열 저장 | 11월 3일 | 7 | 11월 10일 | 완료 |
| 염기서열의 ORF 저장 | 11월 11일 | 7 | 11월 18일 | 완료 |
| ORF에서 intron 제거 | 11월 19일 | 10 | 11월 29일 | 진행중 |
| CDS를 단백질로 변환 | 11월 30일 | 7 | 12월 7일 |  |
| Kozak strength 계산 | 12월 8일 | 7 | 12월 15일 |  |