

알고리즘 설계 프로젝트

n-Queens 문제 (Backtracking, Genetic Algorithm)

보고서 작성 서약서

1. 나는 타학생의 보고서를 베끼거나 여러 보고서의 내용을 짜집기하지 않겠습니다.

2. 나는 보고서의 주요 내용을 인터넷사이트 등을 통해 얻지 않겠습니다.

3. 나는 보고서의 내용을 조작하지 않겠습니다.

4. 나는 보고서 작성에 참고한 문헌의 출처를 밝히겠습니다.

5. 나는 나의 보고서를 제출 전에 타학생에게 보여주지 않겠습니다.

나는 보고서 작성시 윤리에 어긋난 행동을 하지 않고 정보통신공학인으로서 나의 명예를 지킬 것을 맹세합니다.

2020년 6월 28일

학부 정보통신공학과

학년 3

성명 심규환

학번 12181793



**개요**

## **(1) 전체 코드**

## **(2) Backtracking**

1. **구현상 특징**

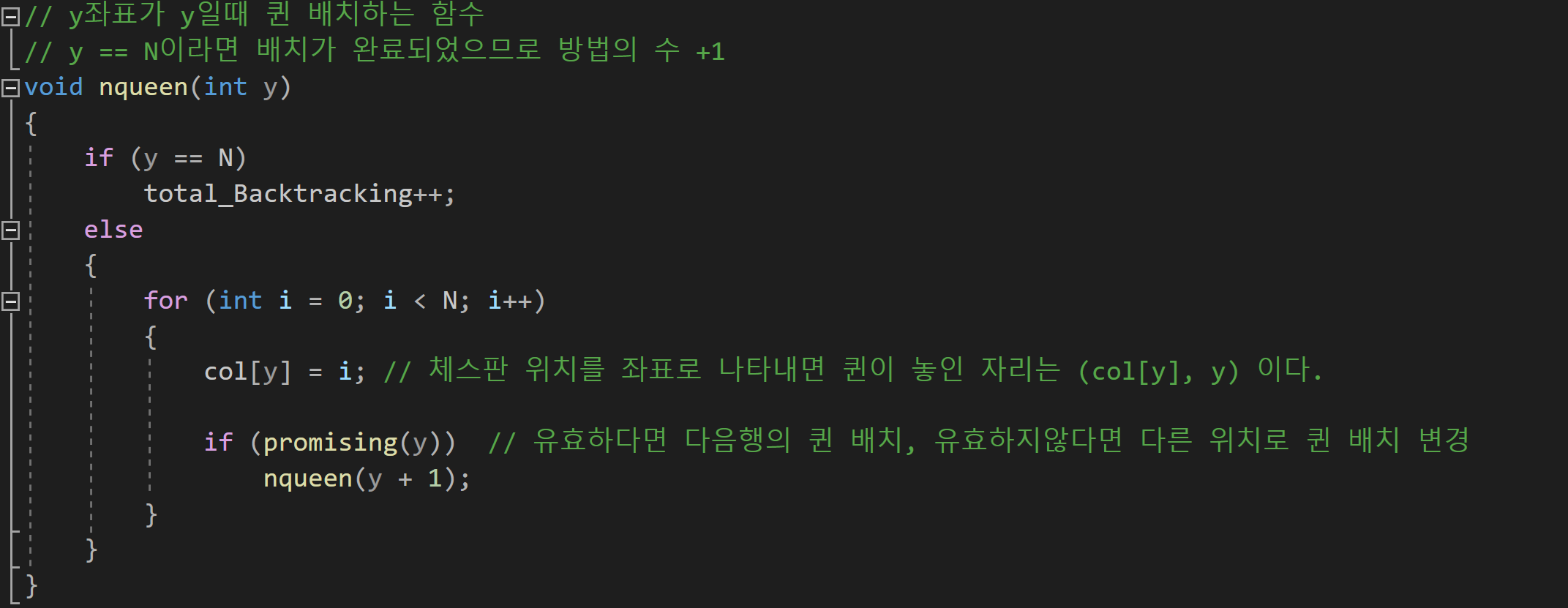
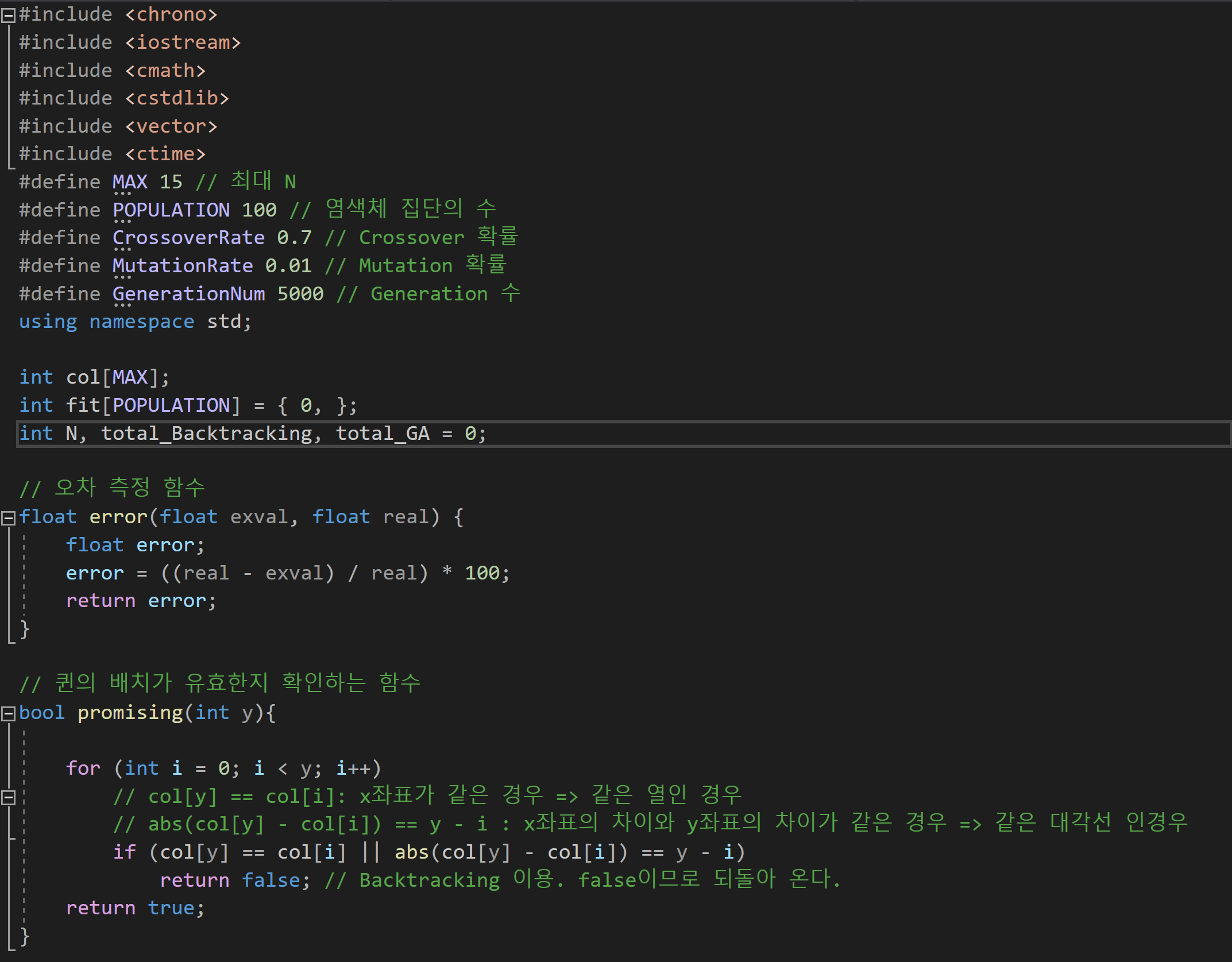
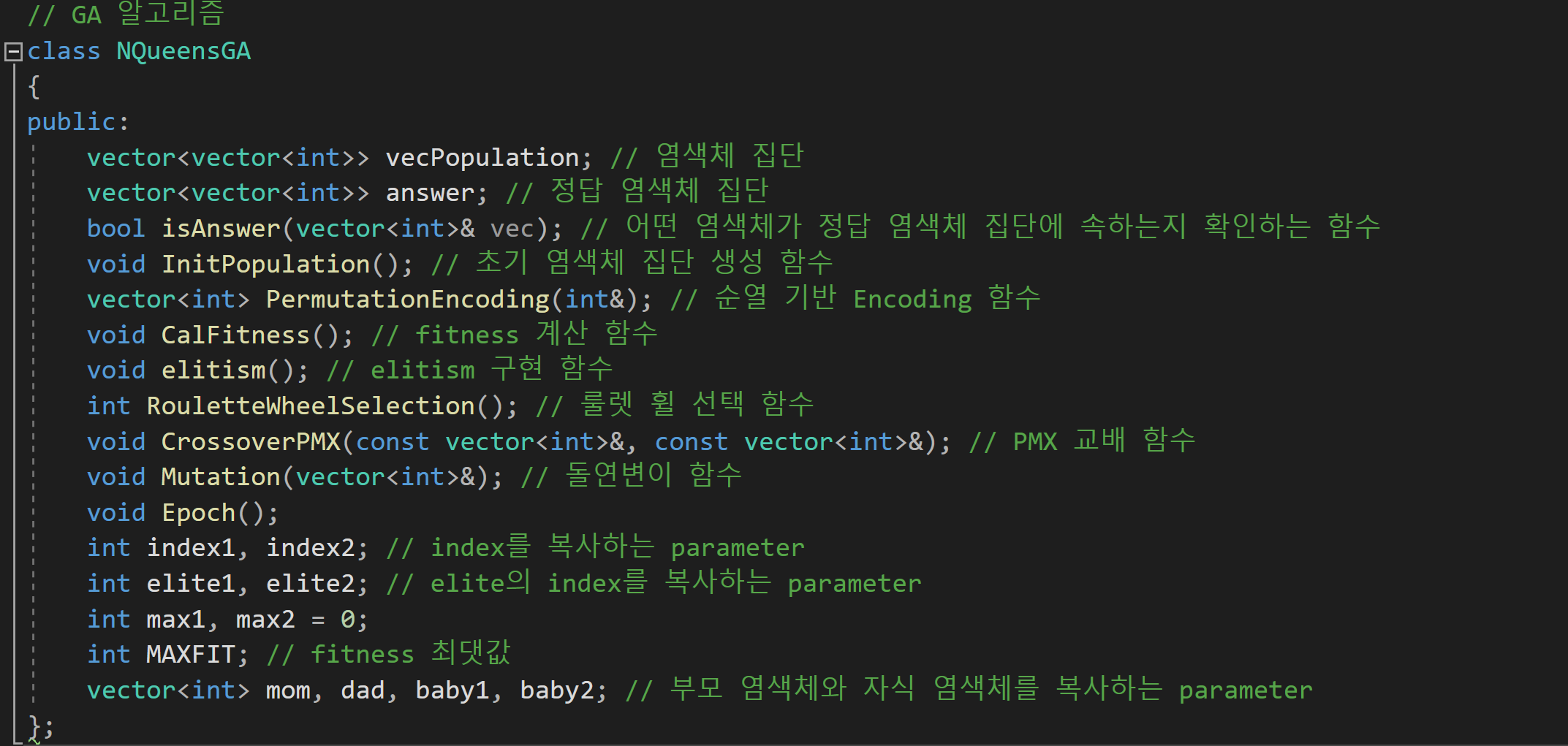
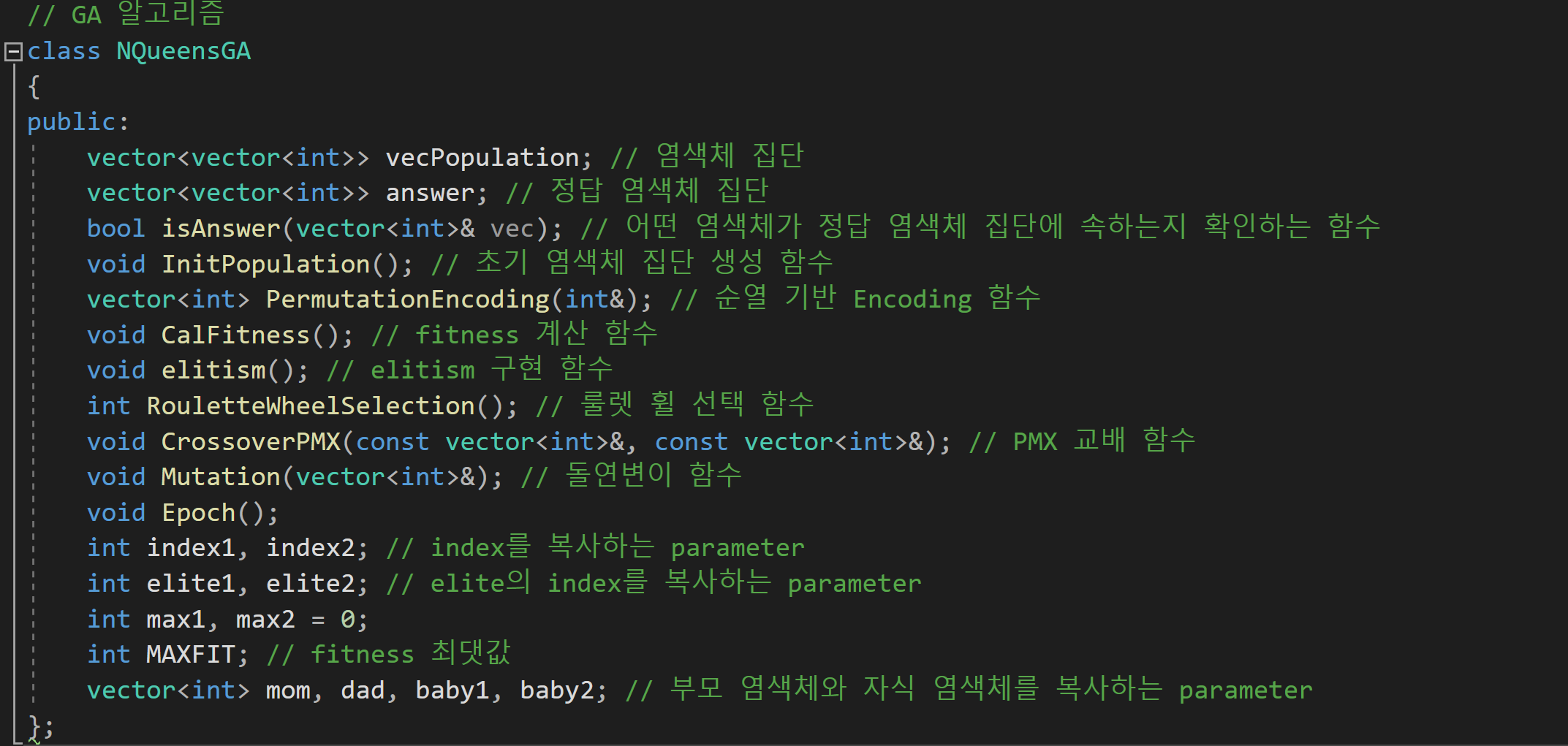
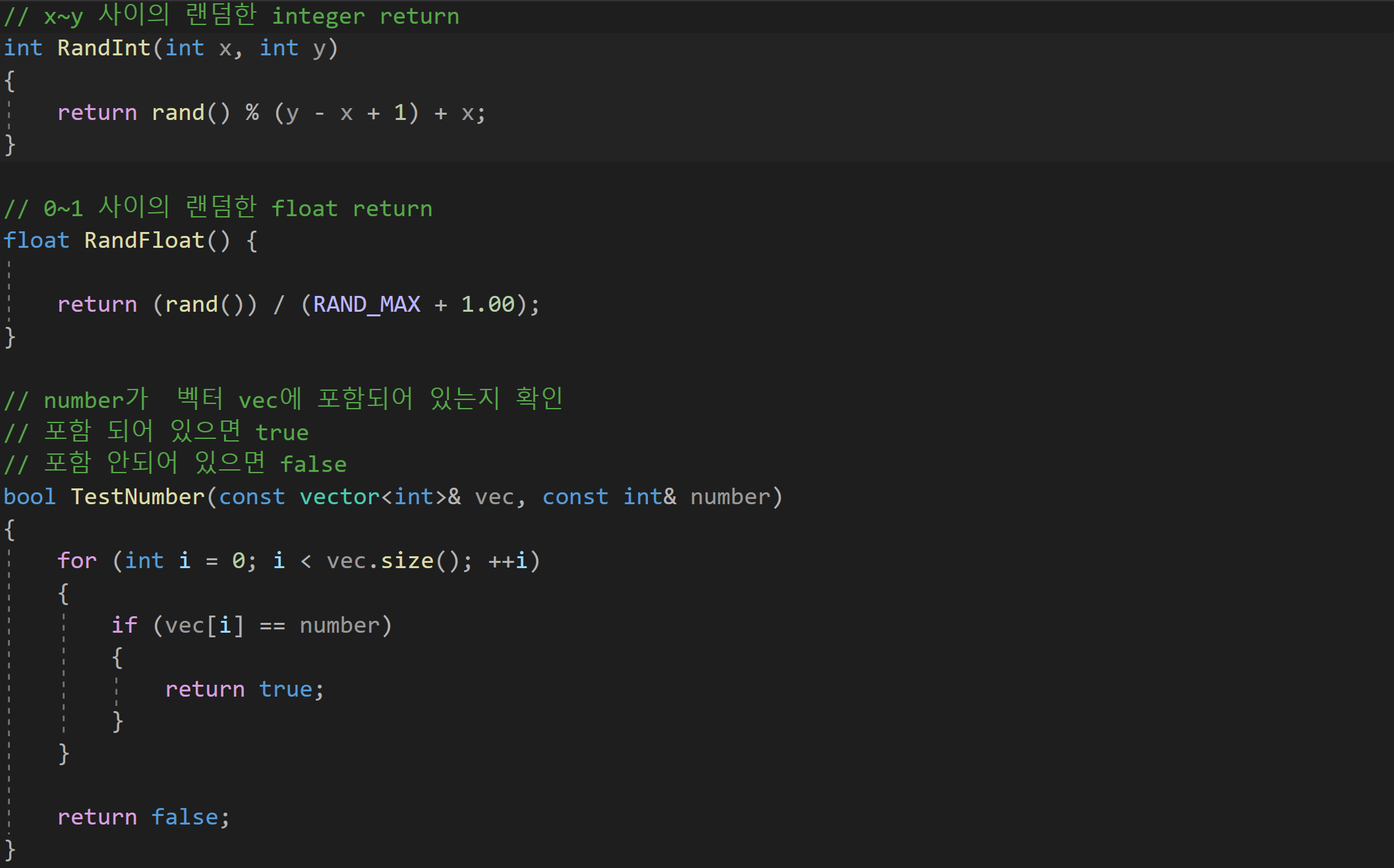
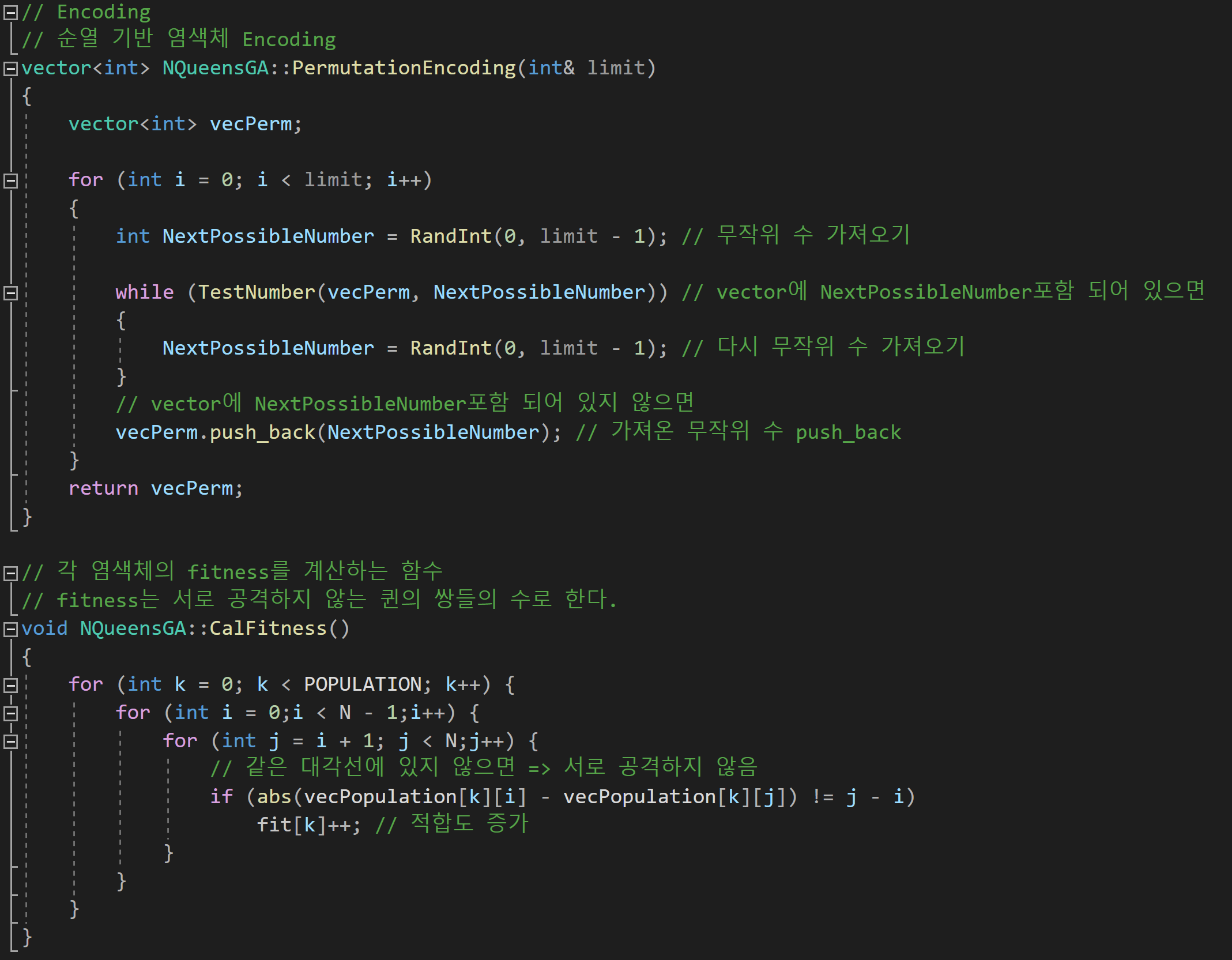
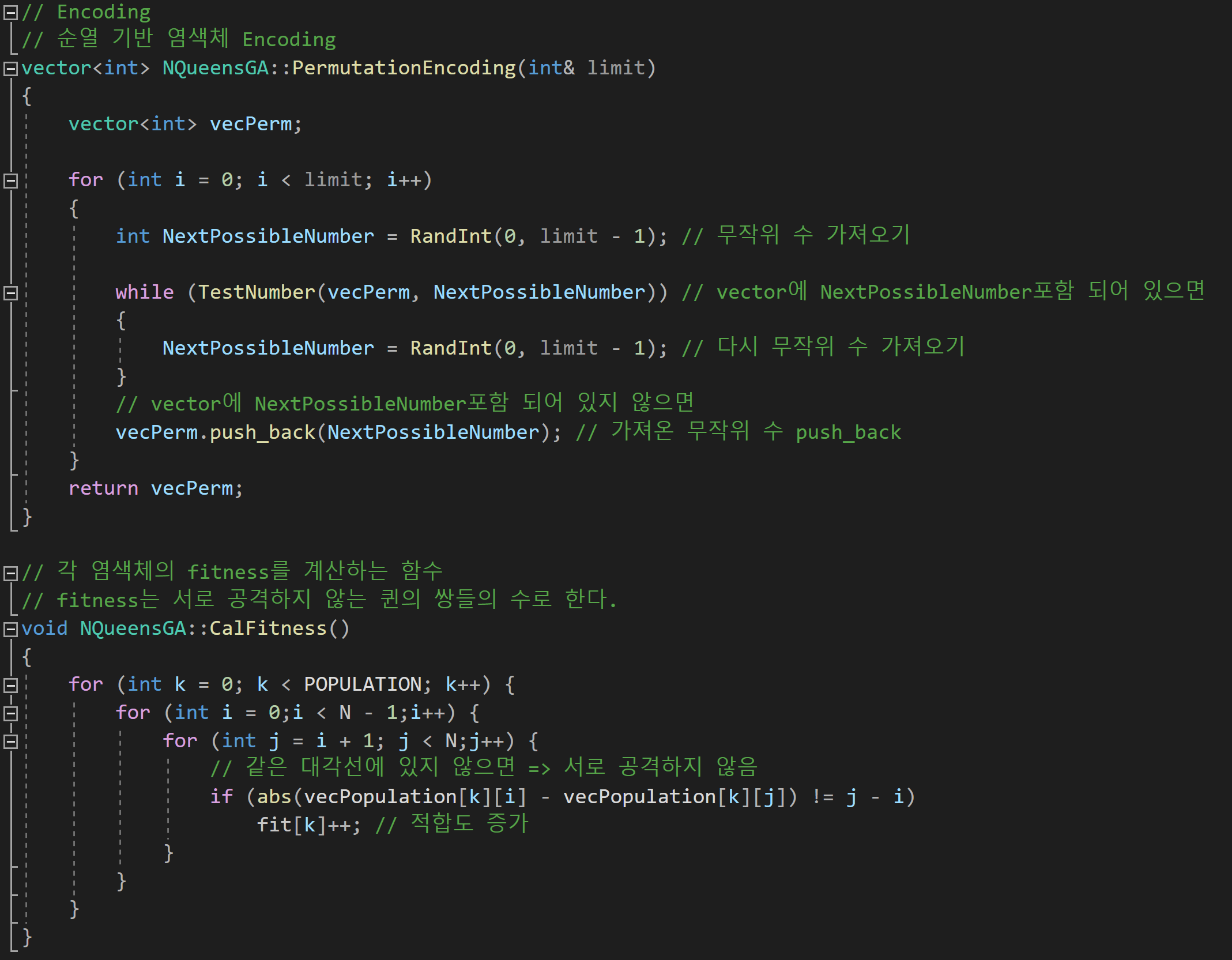
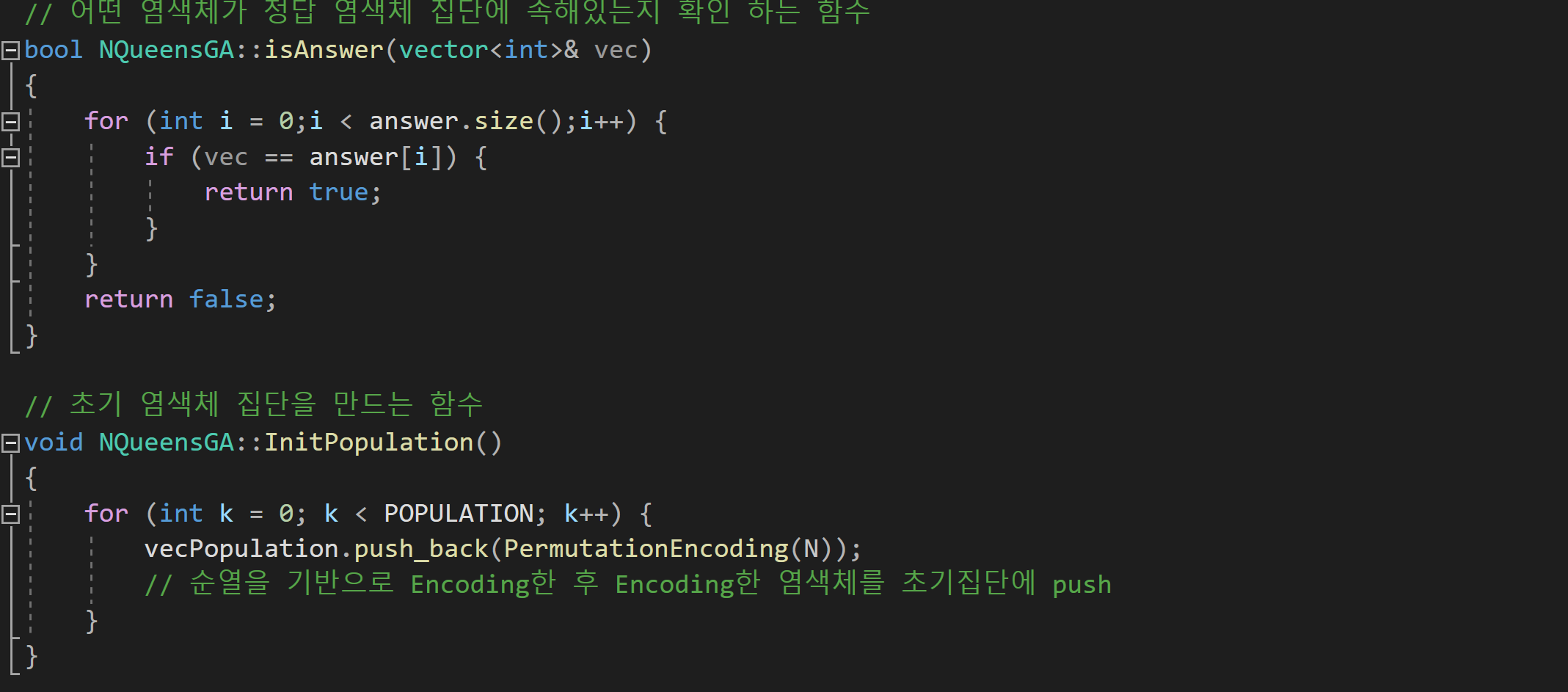
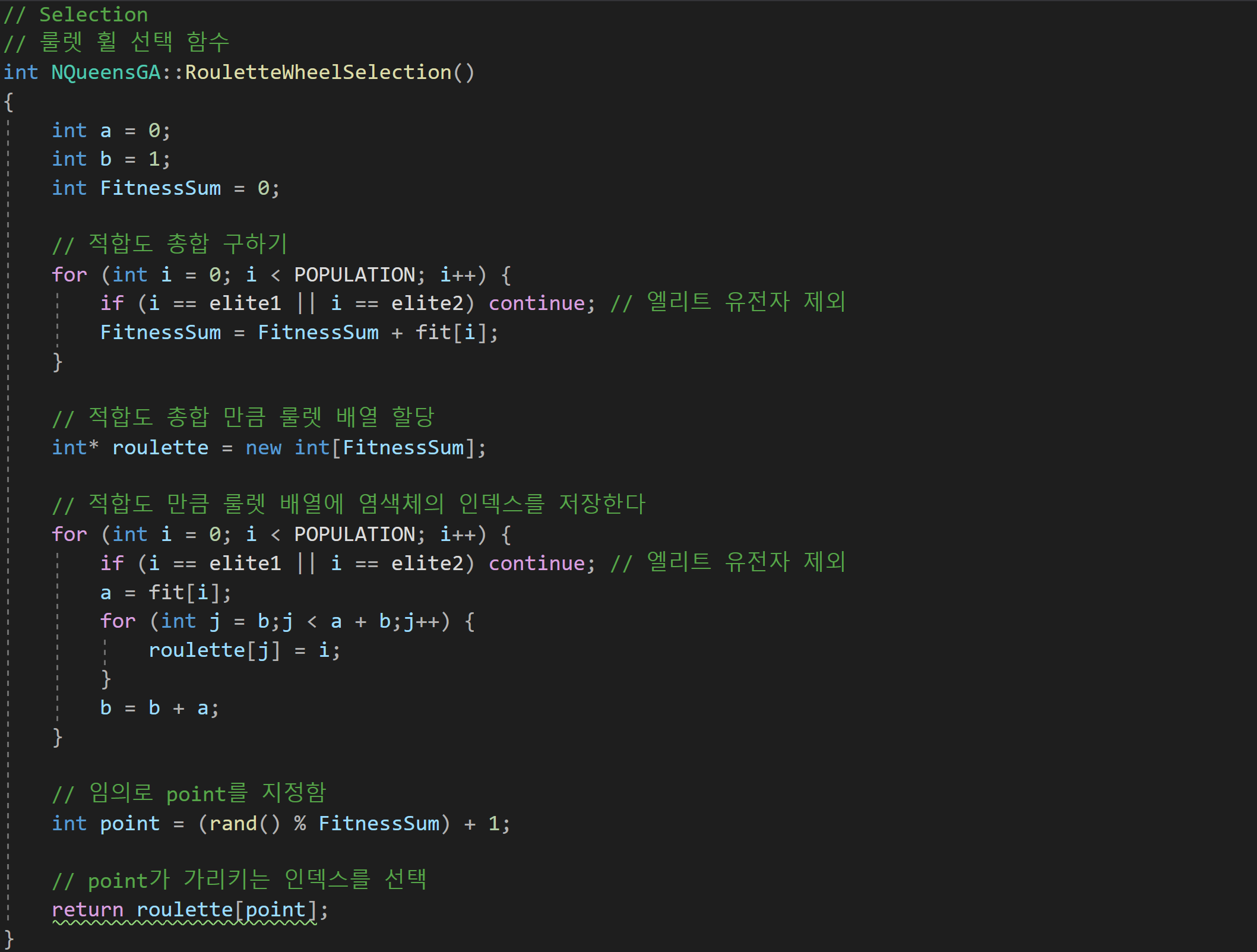
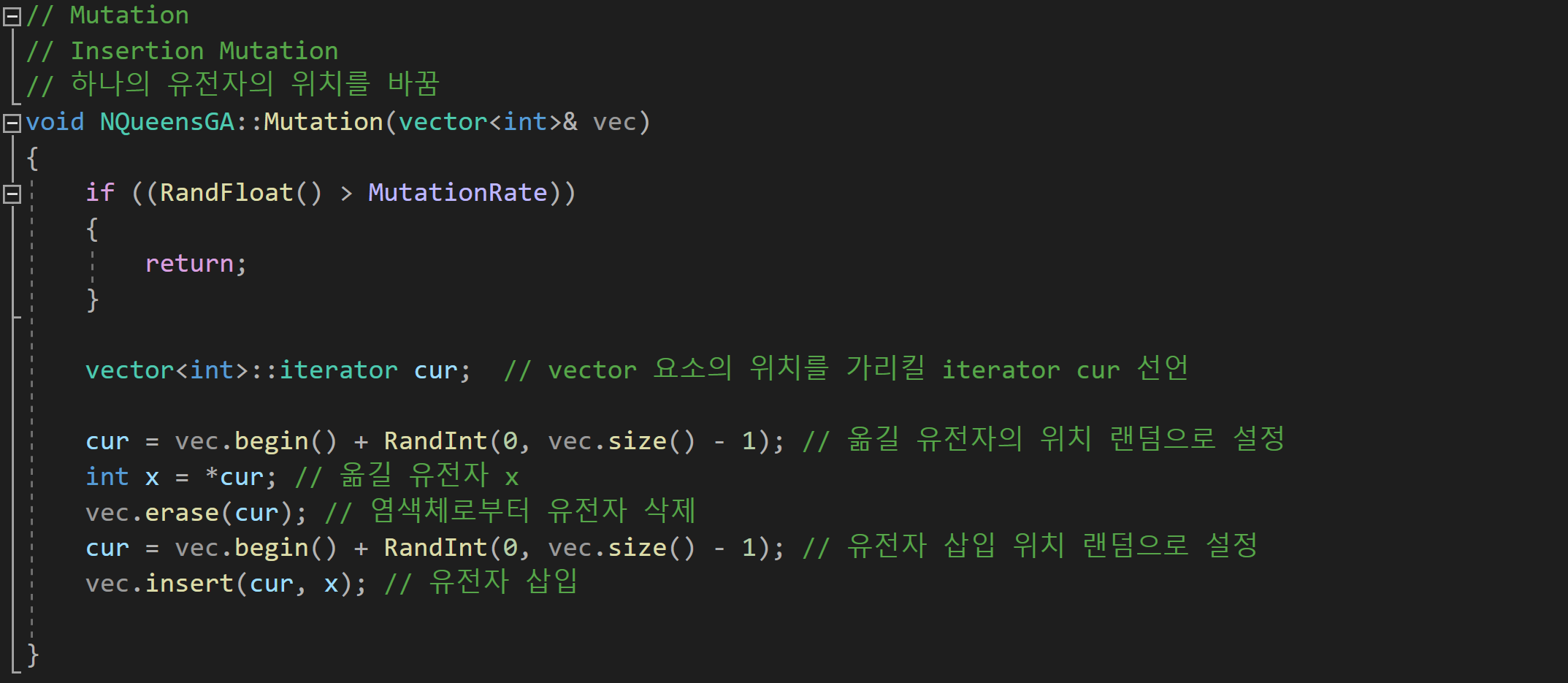
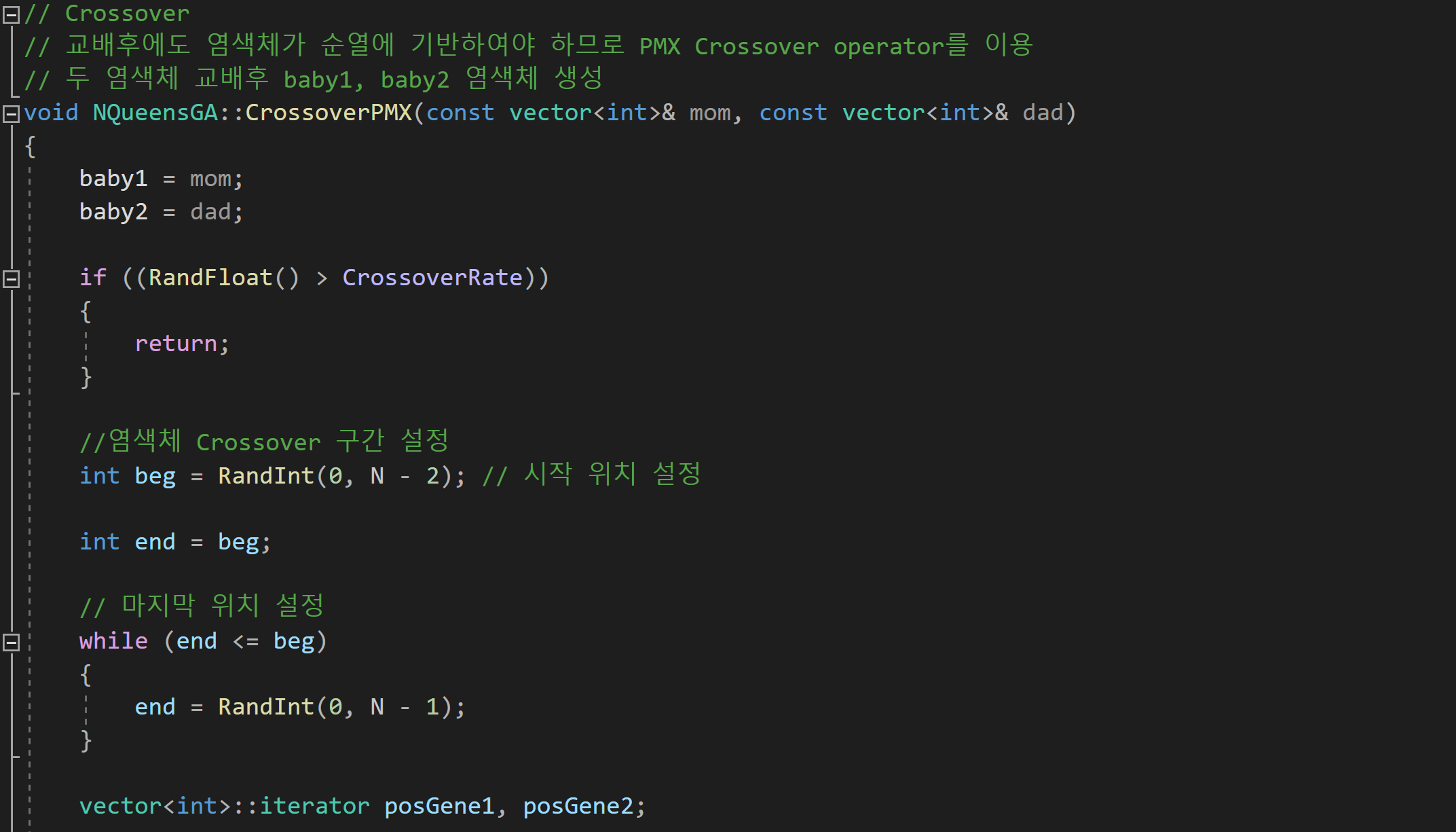
## **(3) Genetic Algorithm**

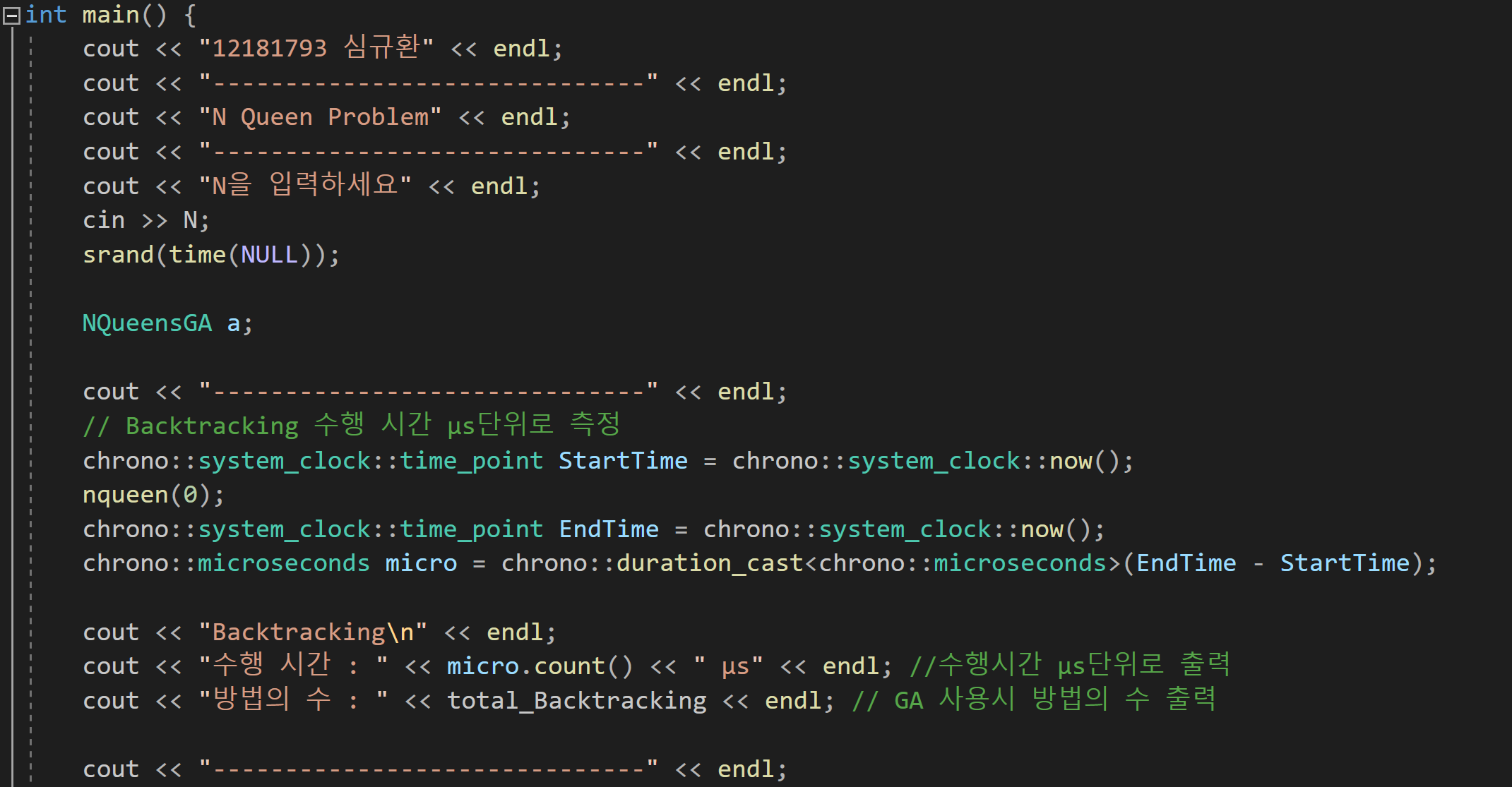
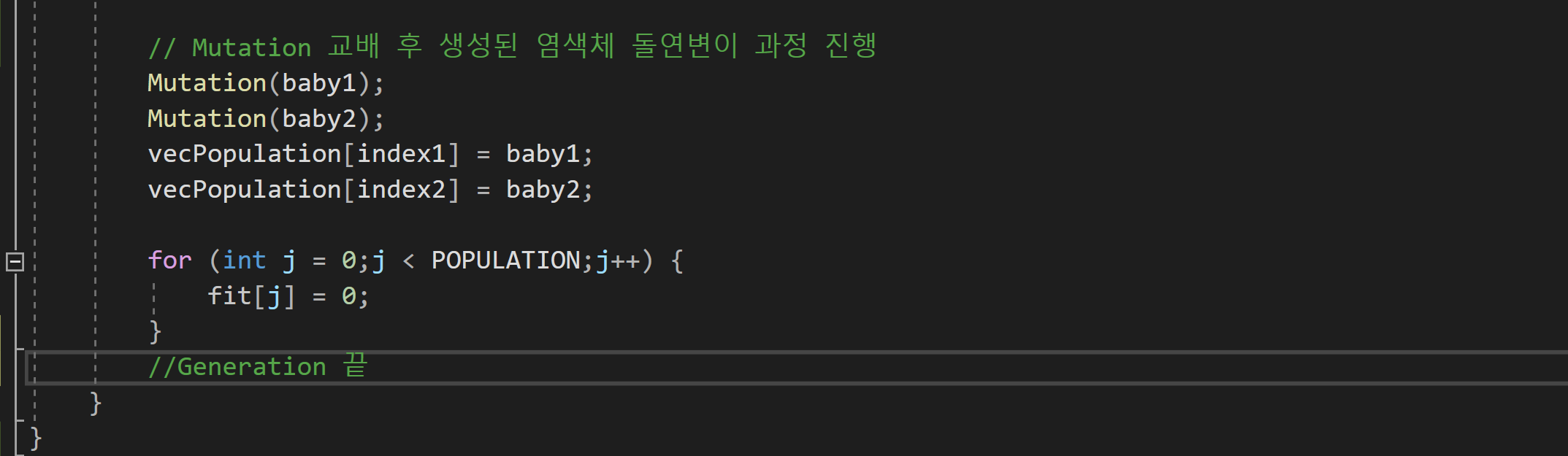
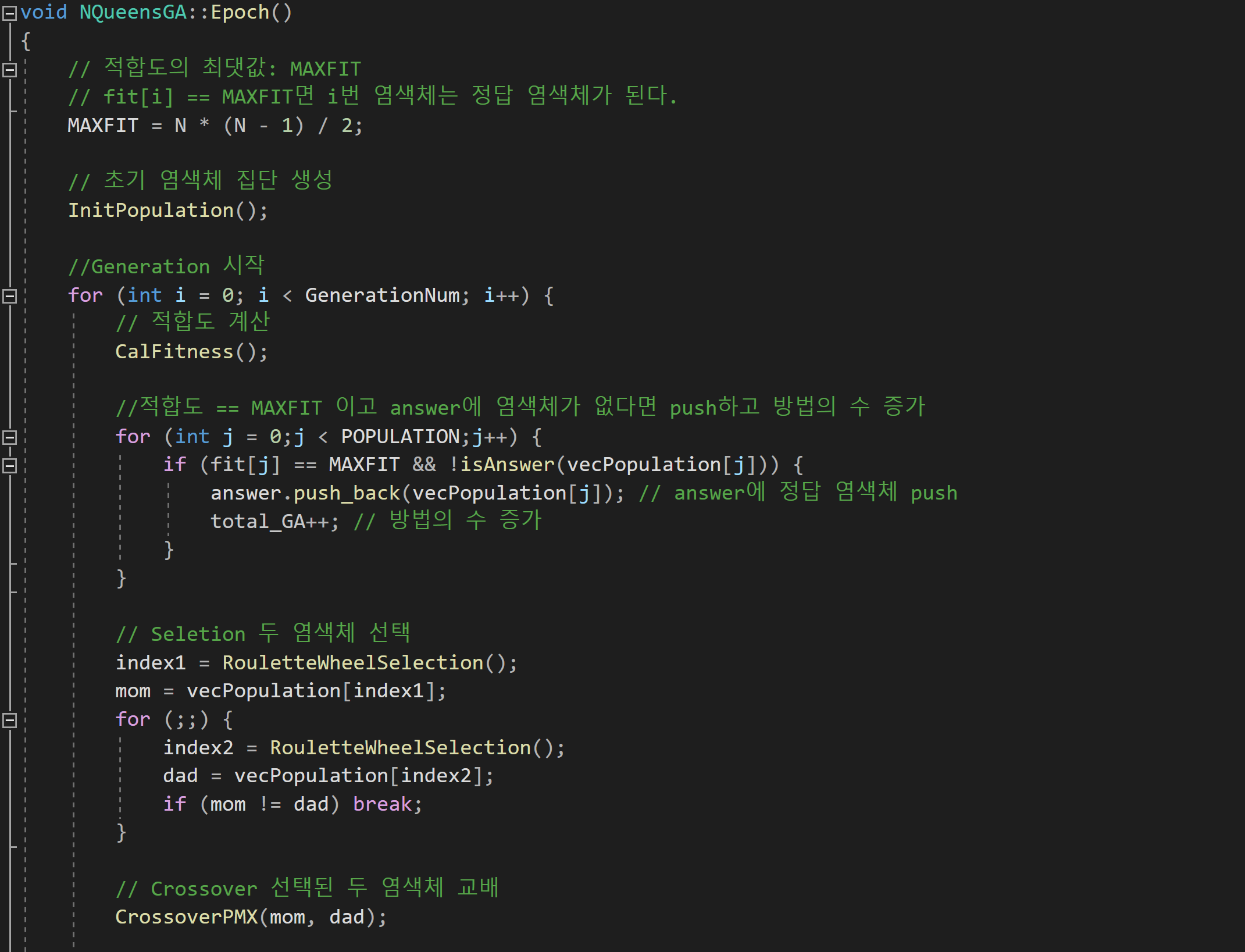
1. **구현상 특징**
   1. **Encoding**
   2. **Selection**
   3. **Crossover**
   4. **Mutation**

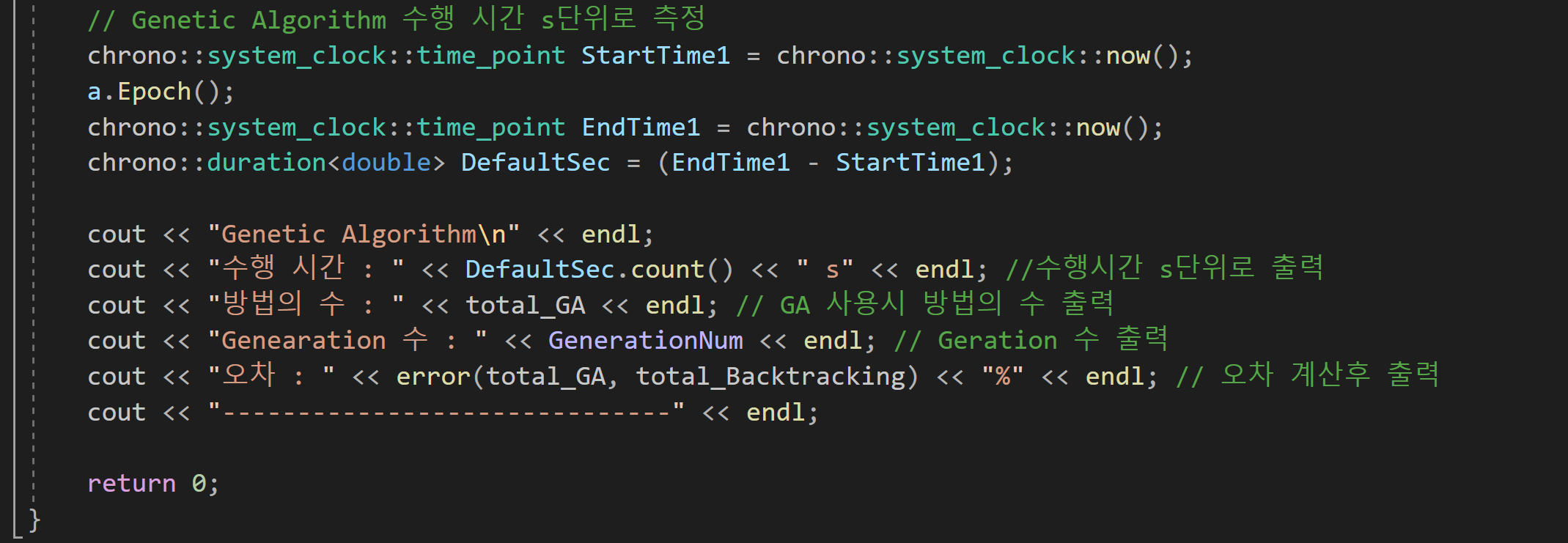
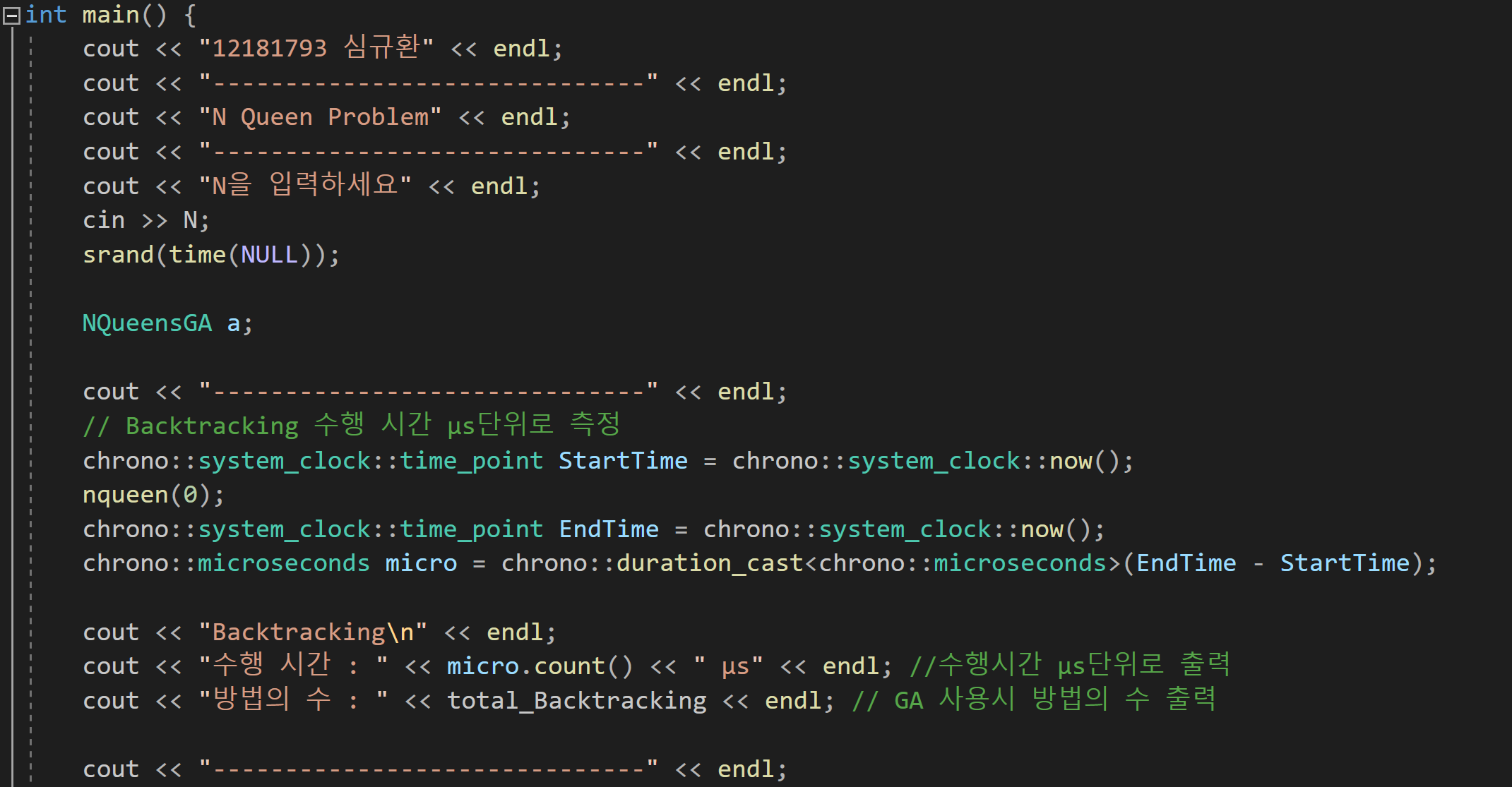
## **(4) 성능 비교 및 분석**

1. **N의 값을 달리하여 비교**
2. **Generation 수를 달리하여 비교**
3. **분석**

**(1) 전체 코드**

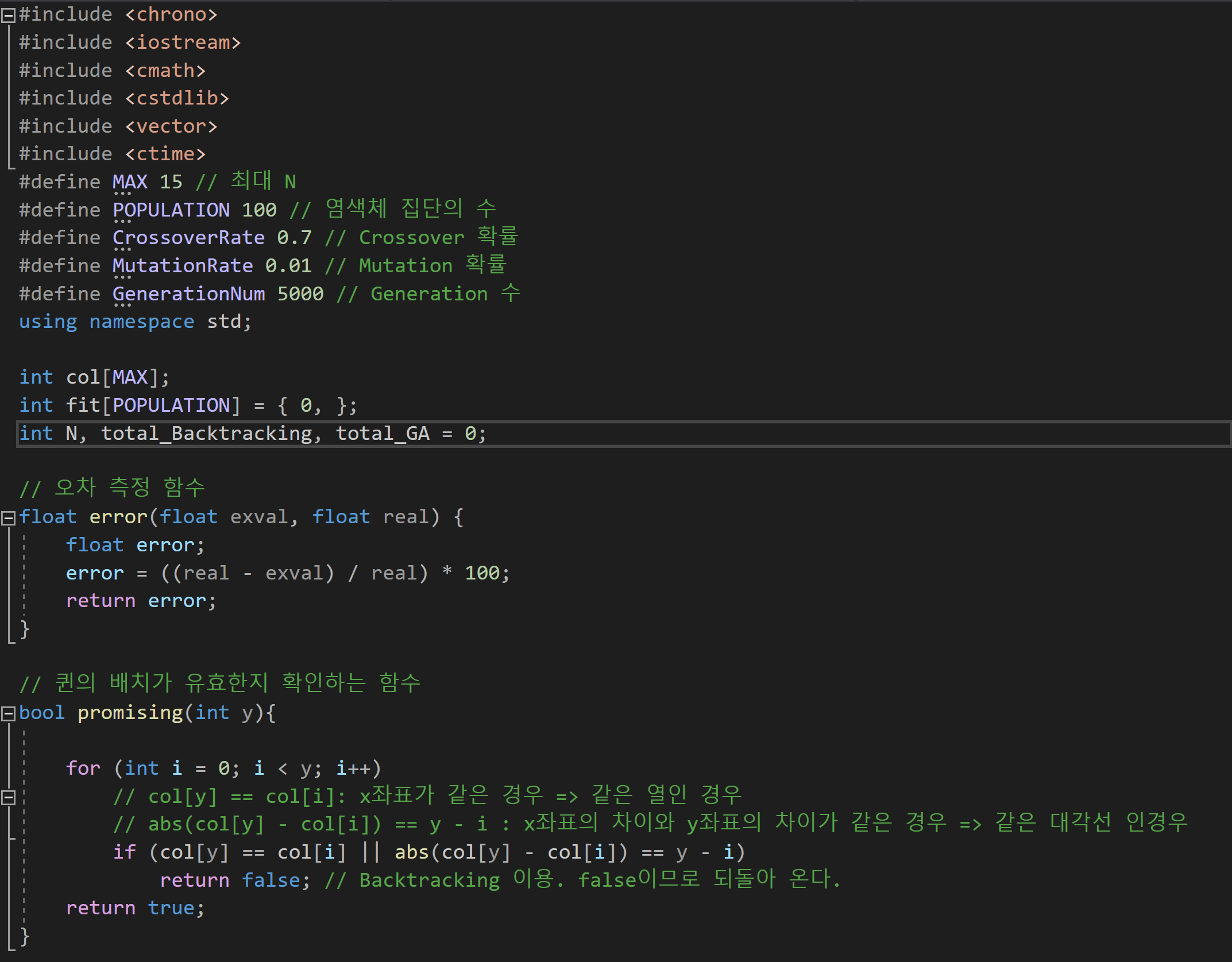
****************

****

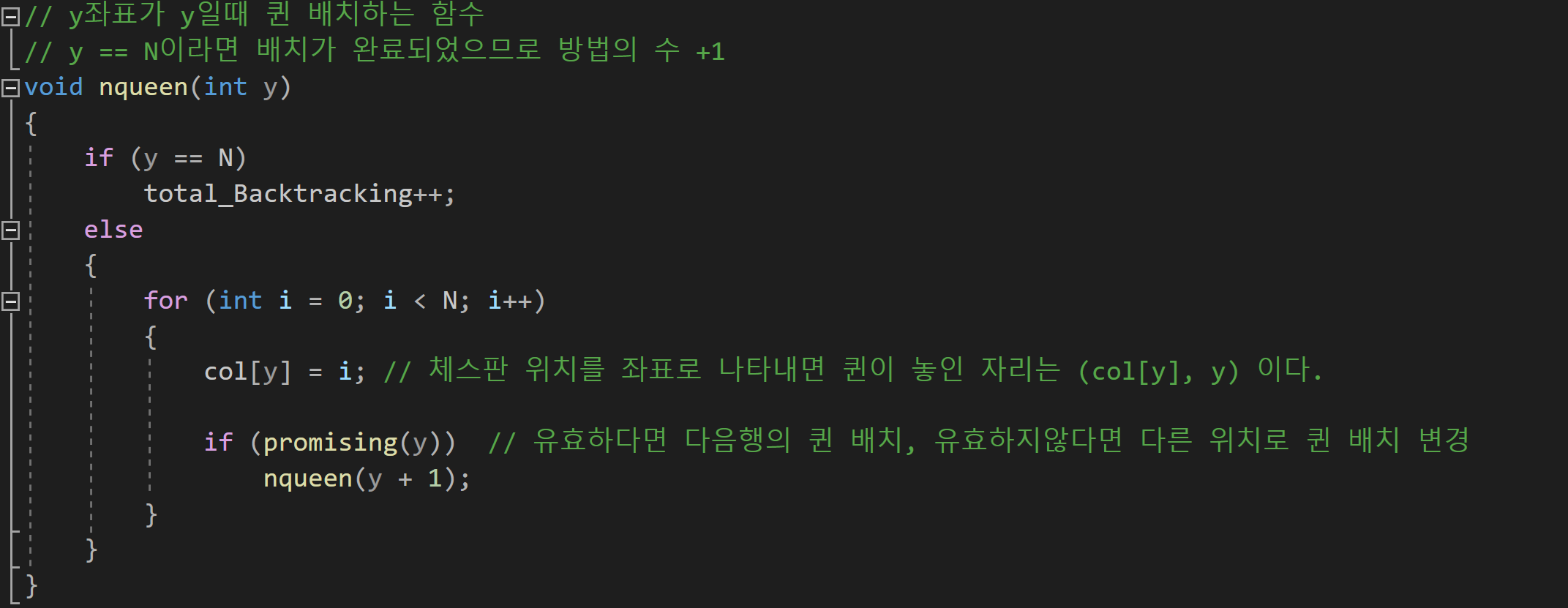
****

**(2) Backtracking**

**i. 구현상특징**

****

**promising 함수는 (col[y], y) 위치의 퀸이 다른 위치의 퀸을 공격하는지 즉 같은 대각선이거나 같은 행, 열인지 확인하고 만약 공격한다면(유효하지 않다면) Backtracking을 한다.**

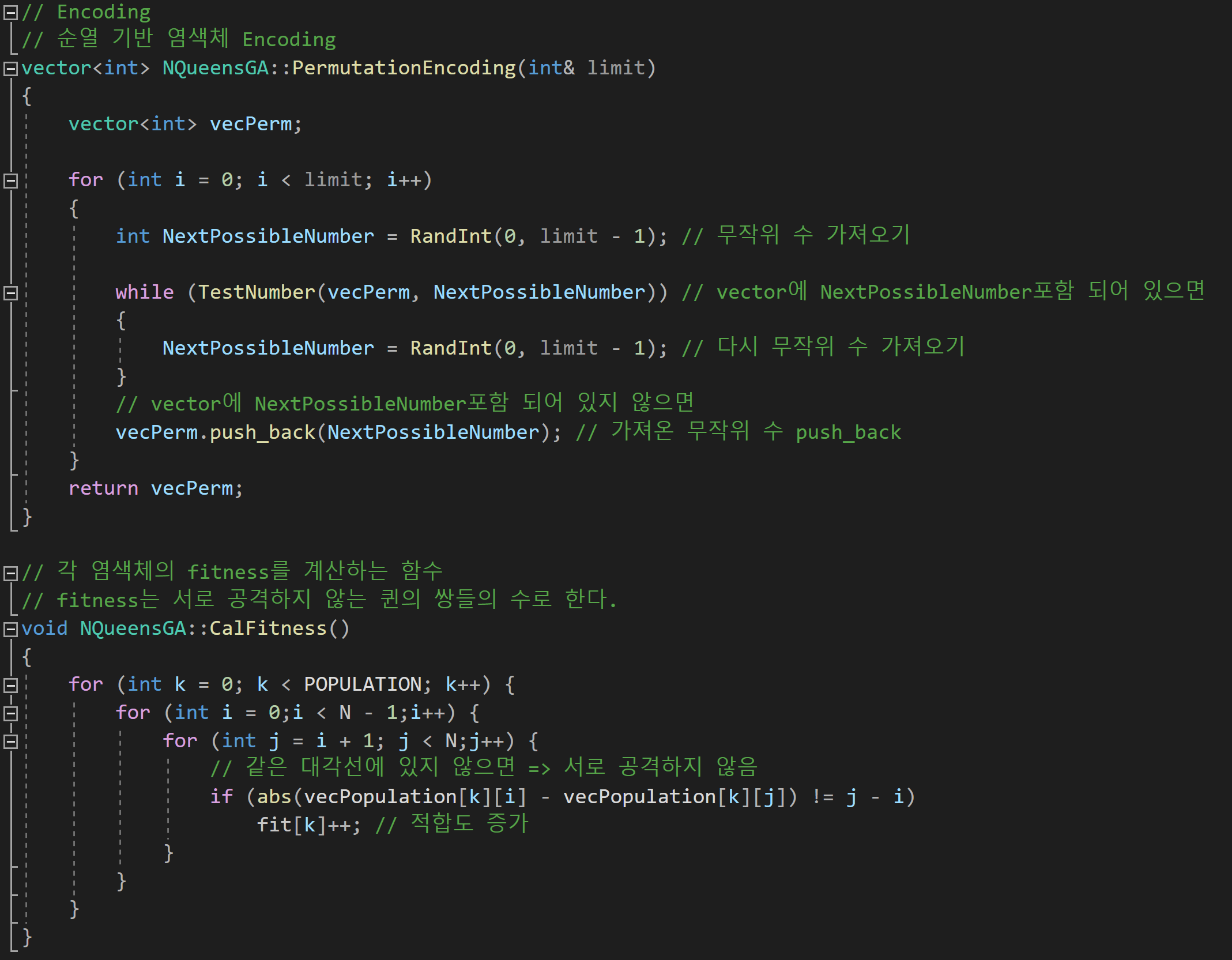
****

**nqueen 함수는 0행부터 하나씩 퀸을 놓고 유효한지 promising 함수로 확인한 다음 유효하다면 퀸을 배치하고 그렇지 않으면 재귀를 통해 퀸의 배치를 변경하도록 한다.**

**(2) Genetic Algorithm**

**i. 구현상특징**

**a) Encoding**

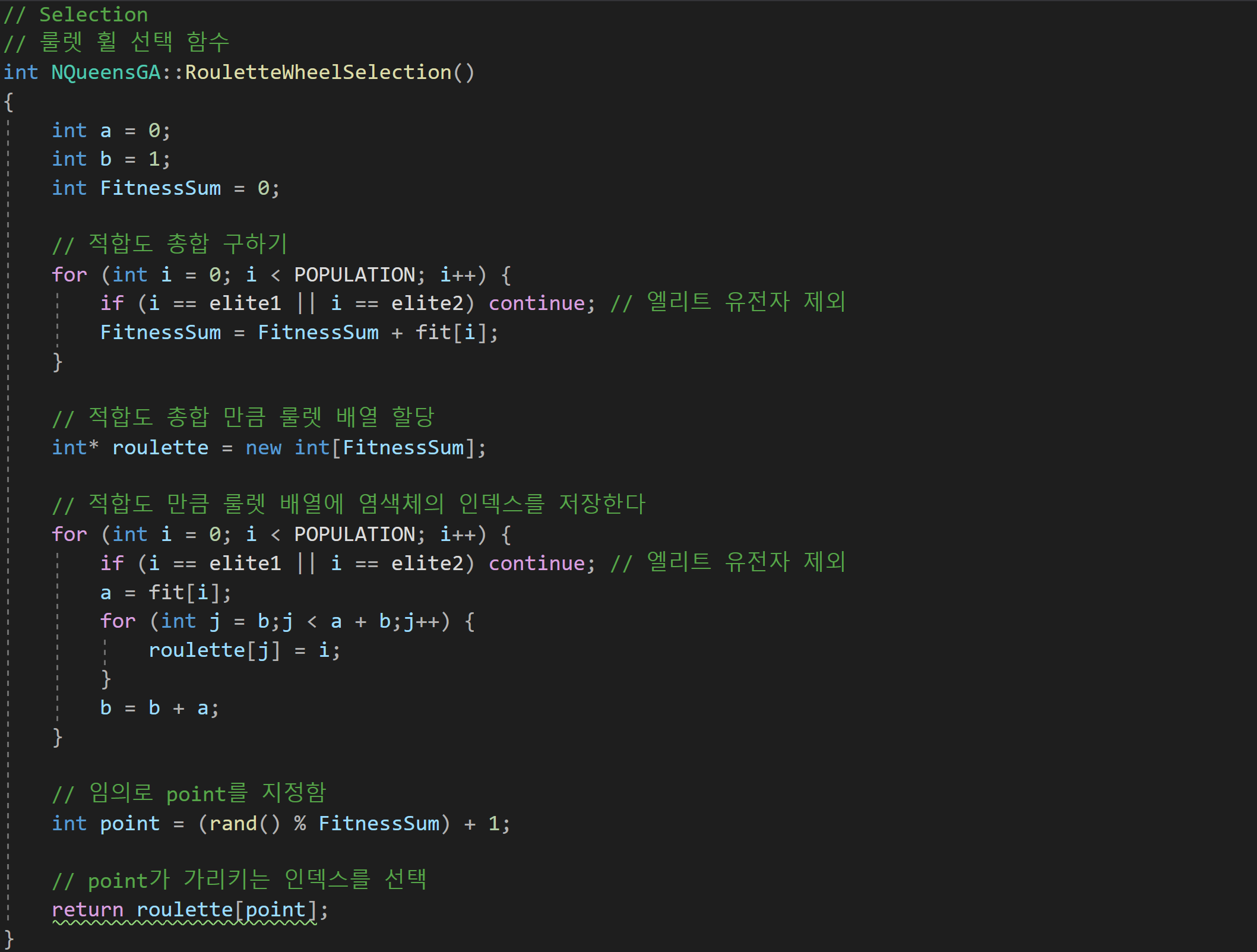
****

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  | **O** |
| **O** |  |  |  |
|  |  | **O** |  |
|  | **O** |  |  |

**vec염색체 (1, 2, 0, 3)**

다음과 같은 퀸의 배치가 있다고 할 때 우리는 염색체 vector<int> vec를 (1, 2, 0, 3)로 설정한다. 즉 vector의 index가 행이 되고 열이 요소가 된다. 이 때 각 요소는 겹치지 않고 0~3 까지의 정수이므로 순열을 기반으로 염색체를 Encoding 한다. 순열 기반으로 염색체를 Encoding하면 다음 과정들을 진행하면서도 염색체가 순열을 기반으로 유지 되어야한다.

**b) Selection**



Seletion은 Elitism을 이용한 Roulette Wheel Seletion을 선택했다. Elite 염색체를 제외한 fitness의 총합을 계산하고 총합 사이즈만큼 배열을 동적으로 할당한다. 그리고 fitness 값만큼 동적으로 할당한 배열에 index값을 저장한다. 그리고 임의로 point를 정하고 point가 가리키는 index를 선택하면 index가 가리키는 염색체를 선택하게 된다. 아래와 같은 경우는 0번 염색체를 선택한다.

**point**

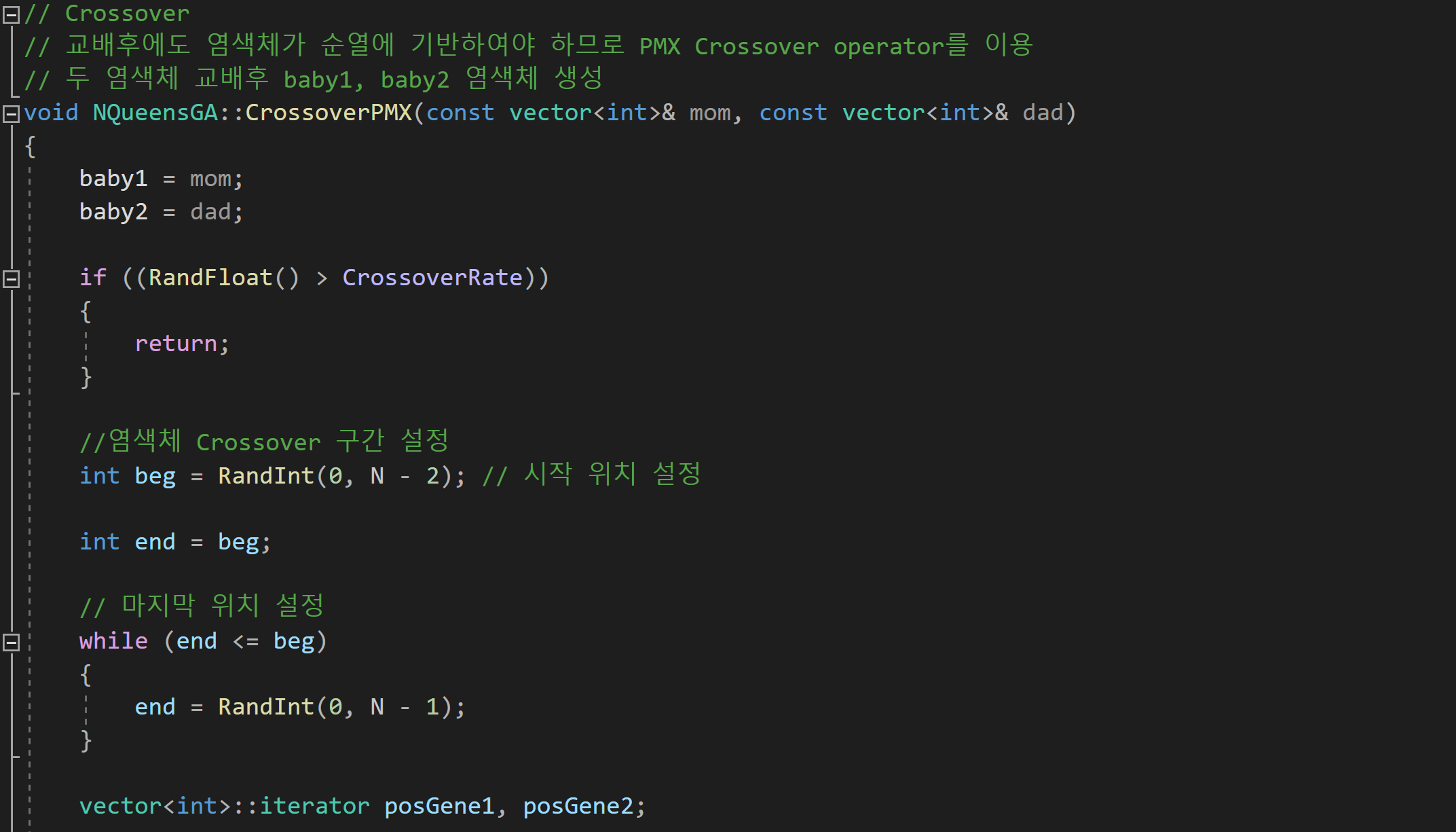
**…**

**2**

**1**

**0**

**c) Crossover**

****

70%의 확률로 교배를 진행하도록 하였다.

순열을 기반으로 Encoding한 염색체를 일반적인 방법으로 교배를 진행하게 되면 교배 후 자손 염색체가 순열을 기반으로 하지 않은 염색체가 나올 수도 있다. 따라서 PMX crossover operator를 이용하여 Crossover를 진행한다. PMX는 임의로 지정한 두 지점에서 치환 규칙을 정하고 한 염색체에서 치환 규칙에 따라 염색체의 유전자 교체를 진행한다.

**↓ ↓**

**염색체1 (2, 5, 0, 3, 6, 1, 4, 7)**

**염색체2 (3, 4, 0, 7, 2, 5, 1, 6)**

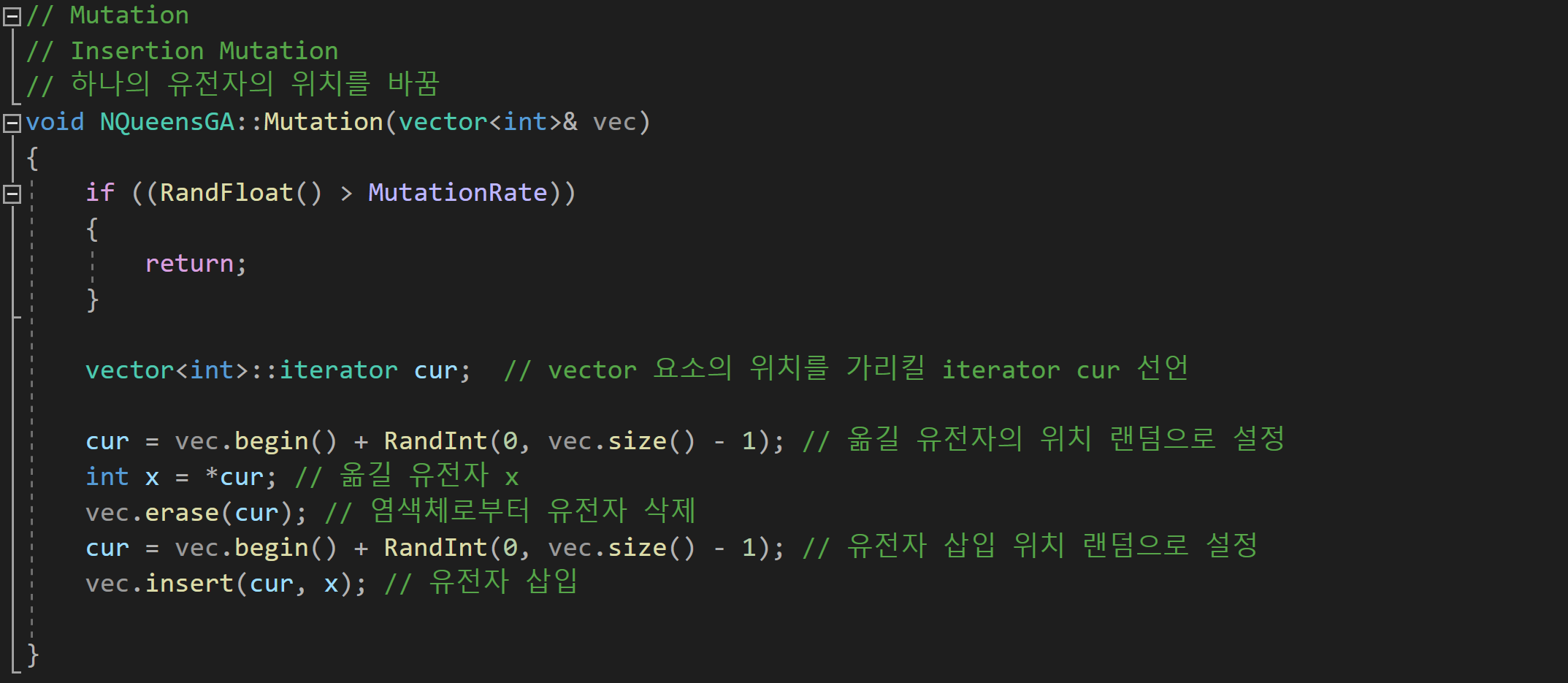
이렇게 임의의 두지점이 선택되면 1과 5를 교체하고, 7과 6을 교체하는 규칙이 생기고

자식 염색체 3과 4가 아래와 같이 생기게 된다.

**염색체3 (2, 1, 0, 3, 7, 5, 4, 6)**

**염색체4 (3, 4, 0, 6, 2, 1, 5, 7)**

**d) Mutation**

****

교배를 진행한 후 1%의 확률로 돌연변이를 생성하게 된다.

Insertion Mutation을 이용하여 돌연변이를 생성하게 만들었다.

Insertion Mutation은 임의의 한 지점을 선정한 후에 그 위치의 유전자를 다른 위치로 이동하도록 하여 돌연변이를 생성한다.

**↓**

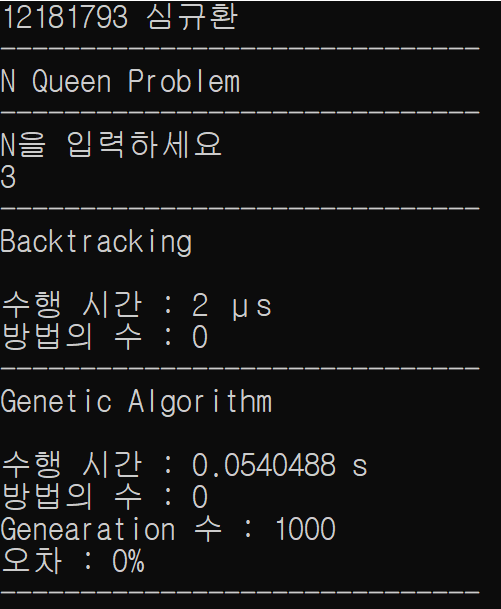
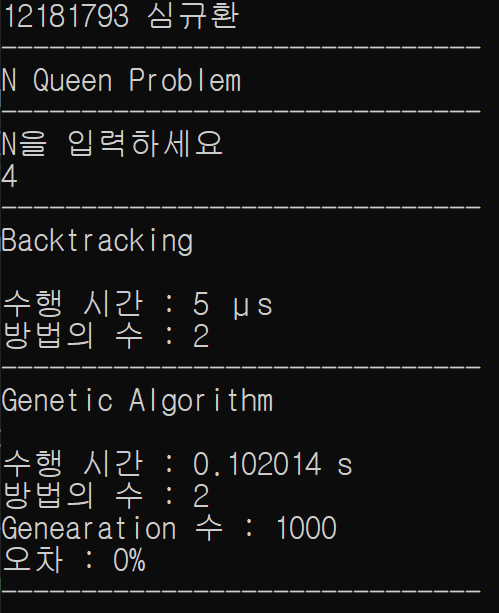
**(2, 5, 0, 3, 6, 1, 4, 7) => 돌연변이 염색체 (2, 5, 1, 0, 3, 6, 4, 7)**

**(4) 성능 분석 및 비교**

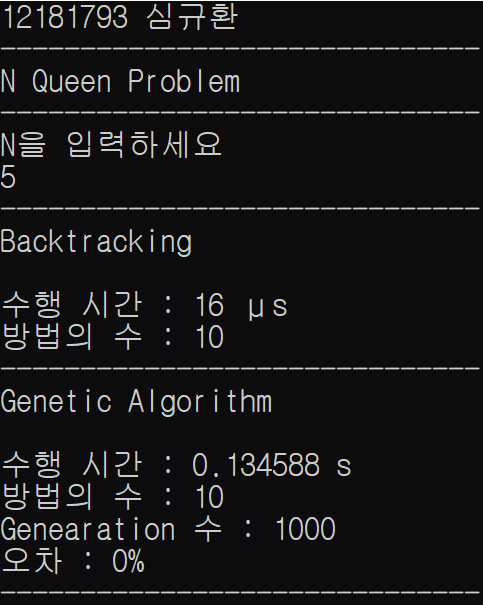
**i. N의 값을 달리하여 비교**

**Generation 수는 1000으로 고정 후 N만 변화시켜가며 비교**

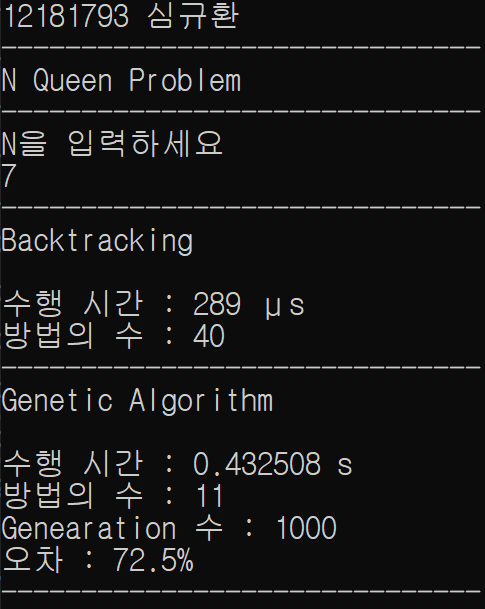
**N=3 N=4**

** **

**N=5 N=6**

** **

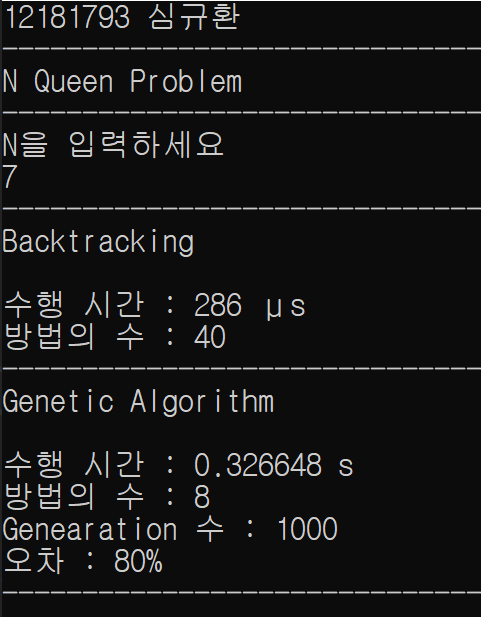
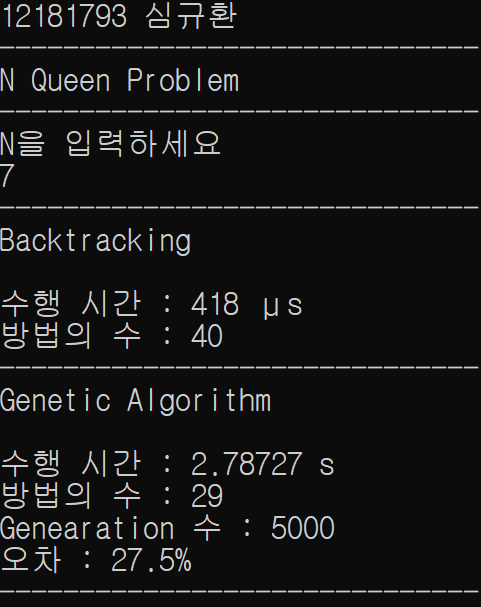
**N=7**

****

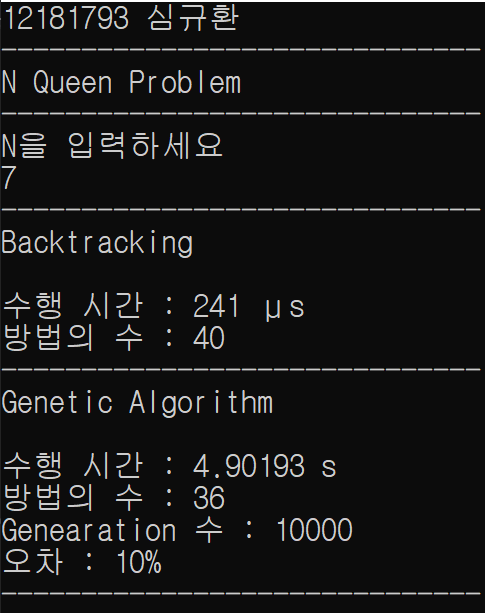
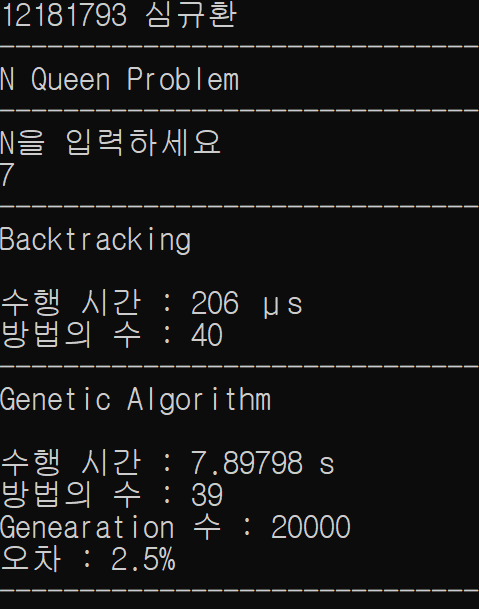
**ii. Generation 수를 달리하여 비교**

**N은 7로 고정 후 Generation 수만 변화시켜가며 비교**

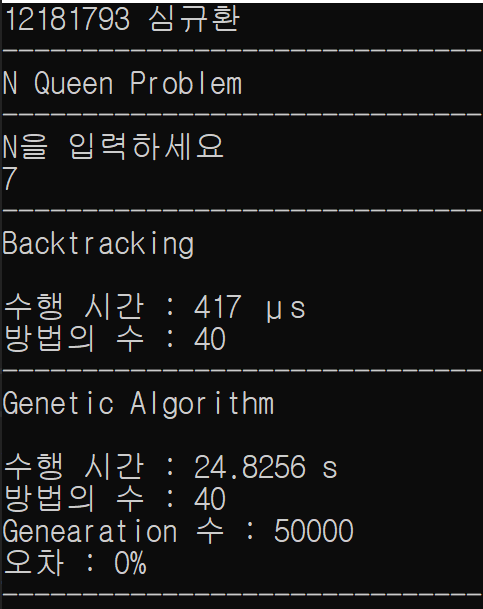
**Generation 수 = 1000 Generation 수 = 5000**

** **

**Generation 수 = 10000 Generation 수 = 20000**

** **

**Generation 수 = 50000**

****

**iii. 분석**

**수행시간 그래프**

**Genetic Algorithm 오차 그래프**

Backtracking의 수행시간 그래프를 살펴보면 N이 커질수록 수행시간이 증가하는 것을 볼 수 있다. 그리고 Generation의 수와는 관계없이 Backtracking이 작동하므로 N=7일 때 약 200 μs ~ 450 μs 내에서 동작함을 추측할 수 있다.

Genetic Algorithm의 수행시간 그래프를 살펴보면 N이 커질수록 수행시간이 증가하며 Generatrion이 커질수록 마찬가지로 수행시간이 증가하는 것을 확인할 수 있다. 오차 그래프를 보면 Generatrion이 커질수록 Genetic Algorithm의 오차가 작아지는 것을 확인할 수 있는데 이를 통해 Generatrion가 많이 지날수록 수행시간은 증가하지만 더욱 정확한 결과값을 얻어 낼 수 있음을 알 수 있다.

Backtracking과 Genetic Algorithm을 비교하였을 때는 Backtracking이 수행시간이 압도적으로 빠를 뿐만 아니라 오차도 없으므로 Backtracking이 압도적으로 성능이 좋은 것을 확인할 수 있다.