Green Algorithm

Team 10

2020312627 권서영

2019312104 김동현

2019310333 남윤성

2018313451 송재현

2018314520 전윤혁

2019314159 조영길

Goal

소스코드 탄소배출량 측정 도구 개발.

탄소배출로 인한 급속한 기후/환경 변화

소프트웨어의 탄소 배출로 인한 환경 부하 가중.
반복적 사용/ 개발/ 유지보수에 의한 탄소 발생 누적

개발자가 작성한 코드에 대한 탄소 발생량 측정 필요

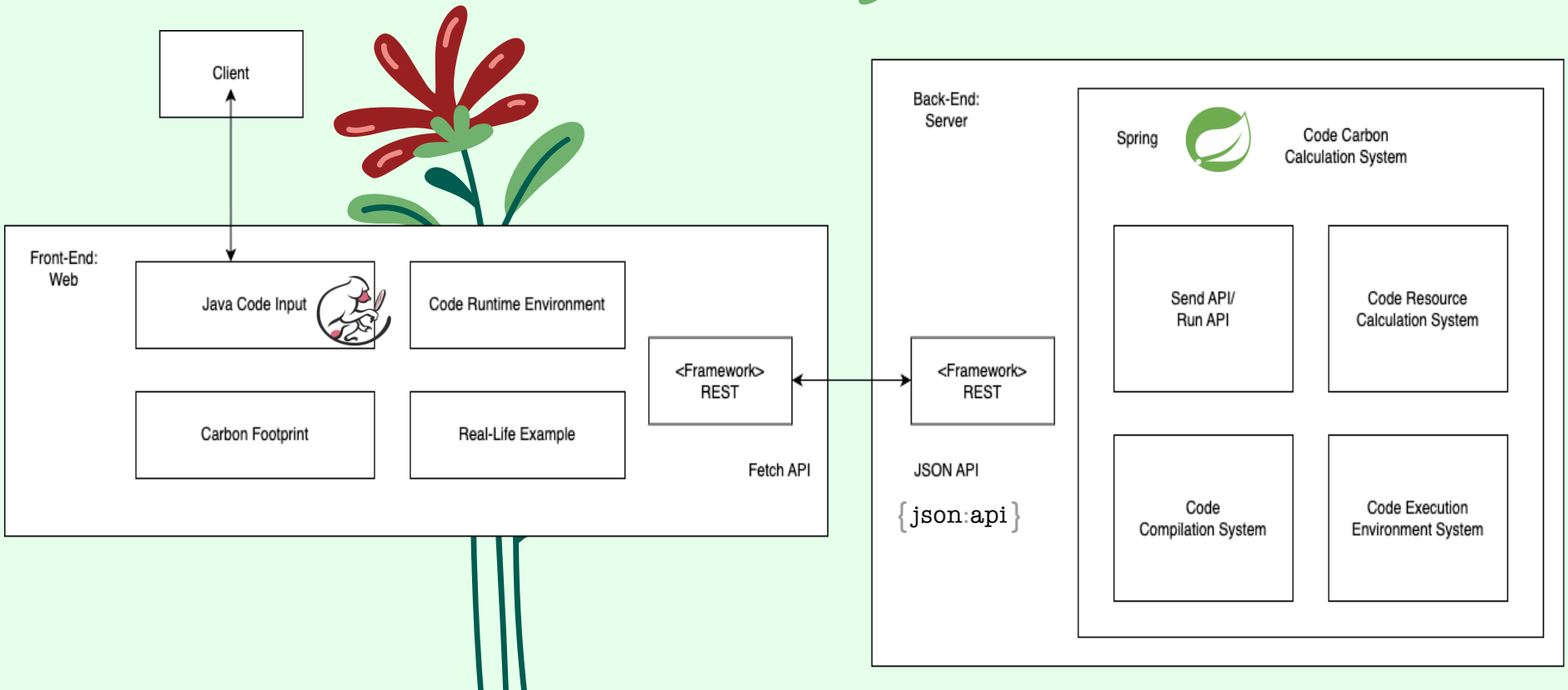
Outline

목차

1. 시스템 구성
2. 구현 결과물 & 시나리오
3. 그림화 패턴 수집 방법
4. 팀 프로젝트 진행 방식

시스템 구성

크게 두 파트로 구성: 프론트 & 백엔드



오픈소스 활용

https://github.com/GreenAlgorithms/green-algorithms-tool/data/latest

GitHub

- Reference Hardware에서 tree, car, train등 변환값, 메모리파워
- Defaults_pue.csv 에서 디폴트 pue
- Tdp_cpu.csv에서 cpu 코어개수, 파워
- CI_aggregated 나라별 carbonintensity

구현 결과물


Green-Algorithm
1
submit
Server Data
Runtime:
Type of cores:
CPU Model:
CPU Number of cores:
CPU power:
CPU usage:
Memory:
Memory power:
PUE:
0 (gCO2e/kWh)
0 (kWh)
0 (hour)
0 (days)
PSF:
Country:
OS name:
OS version:
Java Version:
0 (Km)
0 (Km)
Cores VS Memory
Memory
Core
Memory, CPU carbon emission
Mem
65
CPU
75

시나리오

그린 알고리즘

Runtime > 10s

Runtime goes out of bound -> error

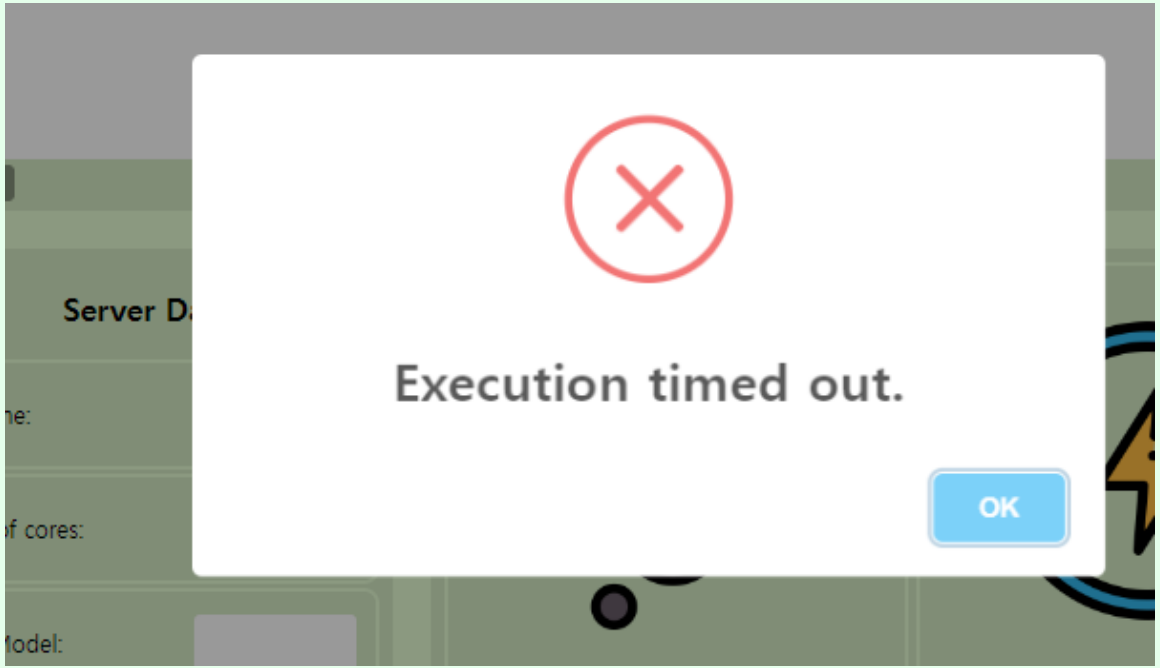
실행되지 않는 코드

Compilation failed -> error

실행 잘 되는 코드

Ideal scenario

시나리오 #1 (런타임 아웃)



오류 출력

시나리오 #2 (컴파일 에러)

오류 출력

시나리오 #3 (정상 실행)

Green-Algorithm

```java
public class PrimeNumberFinder {
    public static void main(String[] args) {
        int targetIndex = 5000;
        int primeCount = 0;
        int number = 2;

        while (true) {
            if (isPrime(number)) {
                primeCount++;
                if (primeCount == targetIndex) {
                    System.out.println(targetIndex + "번째 소수: " + number);
                    break;
                }
            }
            number++;
        }
    }

    private static boolean isPrime(int num) {
        if (num < 2) {
```

submit

실행 가능한 코드

시나리오 #3 (정상 실행)

```
                    break;
                }
            }
            number++;
        }
    }

    private static boolean isPrime(int num) {
        if (num < 2) {
            return false;
        }
        for (int i = 2; i <= Math.sqrt(num); i++) {
            if (num % i == 0) {
                return false;
            }
        }
        return true;
    }
}
```

running

실행 중 일 때 running이 뜨며
코드 수정 및 submit 불가

시나리오 #3 (정상 실행)

Server Data

Runtime:	0.718 (sec)
Type of cores:	CPU
CPU Model:	Core i7-10700K
CPU Number of cores:	8
CPU power:	15.6
CPU usage:	1
Memory:	1614103176
Memory power:	0.3725
PUE:	1.67
PSF:	1
Country:	Korea
OS name:	Windows 10
OS version:	10.0
Java Version:	18.0.2.1

62.4705 (gCO2e/kWh)

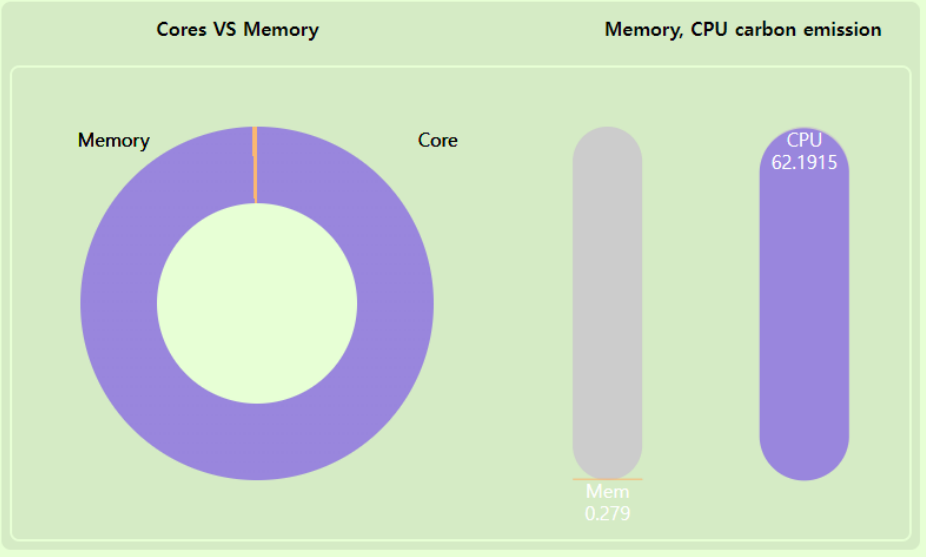
0.1503 (kWh)

2.505 (hour)

2.0445 (days)

0.2489 (Km)

1.5237 (Km)

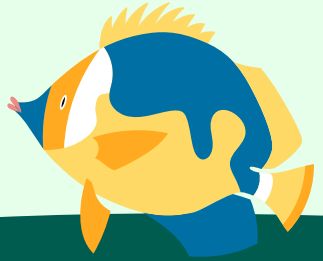


정상적 실행 후 결과값

그린화 패턴 수집 방법

01	Memory access	Reduce unnecessary memory allocation, and utilize better cache affinity & spatiality.
02	Runtime	Codes that reduce runtime
03	code optimization	Consider assembly and machine code level and optimize codes accordingly.

1. Memory Access



cache Spatial Locality

01 cache miss

How can I reduce cache miss?

1-1 2D- array locality

1-2 ArrayList vs. LinkedList

1. Memory Access

Memory Allocation

02 Reduce memory usage

2-1 반복문 & 지역 변수 선언 최적화

2-2 효율적인 변환 작업 함수 사용

2-3 효율적인 결합 작업 함수 사용

2. Runtime

Space complexity

반복문 내부 선언

01

02

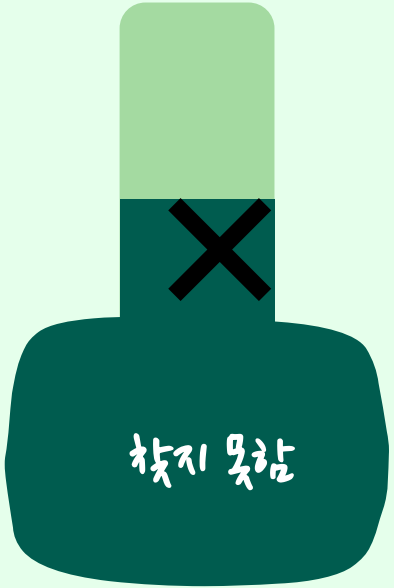
Time complexity

Time complexity는 그린화 패턴을 찾지 못했으나,
이론상으로는 보이어-무어 알고리즘 같은 방법을 사용하면 탄소 배출량을 줄일 수 있다고 판단된다.

3. code optimization

Assembly

Assembly



팀 프로젝트 진행방식- 분담

what to do

- Front-End — 김동현, 남윤성, 조영길
- Back-End — 권서영, 송재현, 전윤혁
- Green Pattern — 모두 다같이~

팀 프로젝트 진행방식- 개발

WEEK	9주차	10주차	11주차	12주차	13주차	14주차
FRONTEND						
BACKEND						
FRONT-BACK 통합						
명세서 작성						
그린화패턴						
TESTING						

감사합니다!!