



프로그래밍언어

[과제01]
prime number

| | |
|-----|------------|
| 과목명 | 프로그래밍언어 |
| 교수명 | 유진 |
| 분 반 | 나반 |
| 학 과 | 컴퓨터학부 |
| 학 번 | 20211799 |
| 이 름 | 유진 |
| 제출일 | 2023.03.22 |

목차

1. C++
2. python
3. JAVA
4. 느낀 점

1. C++

```
Input the number of numbers to process: 11
Input the numbers to be processed: 368 12 58 74 712 12 38 1110 1612 4 222
Number of prime numbers between 4, 12: 3
Number of prime numbers between 12, 38: 7
Number of prime numbers between 38, 58: 4
Number of prime numbers between 58, 74: 5
Number of prime numbers between 74, 222: 26
Number of prime numbers between 222, 368: 26
Number of prime numbers between 368, 712: 54
Number of prime numbers between 712, 1110: 59
Number of prime numbers between 1110, 1612: 68
Total execution time using C++ is 0.009 seconds!
```

```
#include<iostream>
#include<vector>
#include<algorithm>
#include<cmath>
#include <ctime>

using namespace std;

int primeArr(int n){
    vector<bool> arr(n+1);
    int cnt=0;

    arr[1] = true; // 1은 소수가 아니므로 true

    for(int i = 2; i <= sqrt(n); i++){
        if(arr[i]) continue; // 이미 체크 됐으면 continue;
        for(int j = i + i; j <= n; j += i) // i를 제외한 i의 배수들은 소수가 아니다.
            arr[j] = true;
    }

    // 소수 출력
    for(int i = 2; i <= n; i++){
        if(!arr[i]){
            //cout<< i << endl;
            cnt++;
        }
    }
    return cnt;
}

int main(){
    clock_t start, finish;
```

```

double duration;

int N;
cout<<"Input the number of numbers to process: ";
cin>>N;

cout<<"Input the numbers to be processed: "<<endl;
int arr[30];
for(int i=0;i<N;i++)
    cin>>arr[i];

start = clock();
//정렬
sort(arr,arr+N);

//중복제거
int arr2[30];
int i, n = 1;
arr2[0] = arr[0];

int cnt=0;

for (int i = 1; i < N; i++)
    if (arr[i] != arr2[n - 1]) arr2[n++] = arr[i];
for (int i = 0; i < n; i++){
    //printf("%d ", arr2[i]);
    cnt++;
}

printf("\n");

//에라토스테네스의 체
//가장 큰수로 소수 판별이 된 배열을 만들어오기
primeArr(arr2[cnt]);

for(int i=0;i<cnt-1;i++)
    cout<<"Number of prime numbers between "<<arr2[i]<<","<<arr2[i+1]<<":
"<<primeArr(arr2[i+1])-primeArr(arr2[i])<<endl;

finish = clock();

duration = (double)(finish - start) / CLOCKS_PER_SEC;
cout<<"Total execution time using C++ is "<<duration<<" seconds!";

```

```
    return 0;
}
```

- primeArr() 함수

이 함수는 에라토스테네스의 체를 이용하여 1부터 N까지의 수 중에서 소수의 개수를 반환하는 함수입니다. 배열 arr을 생성하고, 이 배열의 인덱스가 소수인지 아닌지를 나타내는 bool 값으로 초기화합니다.

그리고 2부터 \sqrt{N} 까지의 수를 순회하면서, 이전에 체크되지 않은 수에 대해서 그 수의 배수를 모두 체크합니다. 이렇게 체크된 수는 소수가 아닙니다.

배열을 순회하면서 소수인 수들을 카운트한 후 반환합니다.

- main() 함수

N 입력 받기 : 숫자의 개수를 나타내는 N을 입력받습니다.

arr 입력 받기 : N개의 수를 입력받고, 이를 배열 arr에 저장합니다.

중복 제거 및 정렬 : 입력된 수 중 중복된 수를 제거하고, 오름차순으로 정렬합니다.

각 숫자 구간마다 소수 개수 구하기 : 각각의 숫자 구간에서 소수의 개수를 구하기 위해, 입력된 수를 arr2 배열에 중복 제거하여 저장합니다. 그리고 인접한 두 수 사이의 소수 개수를 출력합니다.

실행 시간 측정 : 프로그램의 실행 시간을 측정하여 출력합니다.

2. 파이썬

```
Input the number of numbers to process: 11
Input the numbers to be processed: 368 12 58 74 712 12 38 1110 1612 4 222
Number of prime numbers between 4, 12: 3
Number of prime numbers between 12, 38: 7
Number of prime numbers between 38, 58: 4
Number of prime numbers between 58, 74: 5
Number of prime numbers between 74, 222: 26
Number of prime numbers between 222, 368: 26
Number of prime numbers between 368, 712: 54
Number of prime numbers between 712, 1110: 59
Number of prime numbers between 1110, 1612: 68
Total execution time using Python is 0.005081653594970703 seconds!
```

```

import time

# 소수 판별 함수
def is_prime(N):
    if N < 2: # 1은 소수가 아님
        return False
    for i in range(2, int(N**0.5)+1):
        if N % i == 0:
            return False
    return True

# 사용자로부터 입력 받기
N = int(input("Input the number of numbers to process: "))

nums = list(map(int, input("Input the numbers to be processed: ").split()))

# 실행 시간 측정 시작
start_time = time.time()

nums = list(set(nums)) # 중복 제거
nums.sort() # 정렬

for i in range(len(nums)-1):
    start, end = nums[i], nums[i+1]
    count = sum(is_prime(j) for j in range(start, end+1))
    print(f"Number of prime numbers between {start}, {end}: {count}")

# 실행 시간 측정 종료 및 출력
end_time = time.time()
duration = end_time - start_time
print(f"Total execution time using Python is {duration} seconds!")

```

- is_prime 함수

는 소수를 판별하는 함수입니다. N이 2 이상이고 N의 제곱근까지의 모든 수로 나누어 떨어지지 않으면 소수입니다. 이 함수를 이용하여 각 숫자 범위에서 소수의 개수를 구합니다.

- 그 외

input 함수를 이용하여 사용자로부터 입력을 받습니다. 첫 번째 입력은 처

리할 숫자들의 개수이고, 두 번째 입력은 처리할 숫자들입니다. 입력 받은 숫자들을 list로 저장합니다. list의 set 함수를 이용하여 중복된 숫자를 제거하고, sort 함수를 이용하여 오름차순으로 정렬합니다.

각 숫자 범위에서 소수의 개수를 구하기 위해 nums 리스트의 인접한 숫자 쌍을 차례대로 선택하여 is_prime 함수를 이용하여 소수의 개수를 구합니다. range 함수를 이용하여 각 숫자 범위를 설정합니다. 소수의 개수를 구할 때 sum 함수를 이용하면 간단하게 구할 수 있습니다.

time 모듈을 이용하여 시작 시간과 종료 시간을 기록하고, 두 시간의 차이를 계산하여 실행 시간을 구합니다. 실행 시간을 출력합니다.

3. JAVA

```
Input the numbers to be processed: 368 12 58 74 712 12
Number of prime numbers between 4, 12: 3
Number of prime numbers between 12, 38: 7
Number of prime numbers between 38, 58: 4
Number of prime numbers between 58, 74: 5
Number of prime numbers between 74, 222: 26
Number of prime numbers between 222, 368: 26
Number of prime numbers between 368, 712: 54
Number of prime numbers between 712, 1110: 59
Number of prime numbers between 1110, 1612: 68
Total execution time using Java is 0.010844 seconds!
```

```
import java.util.*;

public class homework1 {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);

        // 입력 받기
        int N = 0;

        System.out.print("Input the number of numbers to process : ");
```

```

N = sc.nextInt();

int[] arr = new int[N];
System.out.print("Input the numbers to be processed: ");
for (int i = 0; i < N; i++) {
    arr[i] = sc.nextInt();
}

// 중복 제거
arr = Arrays.stream(arr).distinct().toArray();

// 정렬
Arrays.sort(arr);

long start = System.nanoTime();
for (int i = 0; i < arr.length - 1; i++) {
    int startNum = arr[i];
    int endNum = arr[i + 1];
    int count = 0;
    for (int j = startNum; j <= endNum; j++) {
        if (isPrime(j)) {
            count++;
        }
    }
    System.out.printf("Number of prime numbers between %d, %d: %d\n",
startNum, endNum, count);
}
long end = System.nanoTime();
double executionTime = (end - start) / 1_000_000_000.0;
System.out.printf("Total execution time using Java is %f seconds!\n",
executionTime);
}

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

```


- isPrime(int n)함수

소수인지 아닌지 판별하는 함수이고 파이썬에서 구현한 방법과 동일하다.

- main 함수

Scanner를 이용하여 사용자로부터 입력을 받습니다. 입력받은 수를 배열에 저장하고 중복을 제거합니다. 그리고 배열을 정렬합니다. 소수 개수 계산을 위해 이전 수와 현재 수를 비교하면서 소수 개수를 세고 출력합니다. 마지막으로 실행 시간을 측정하여 출력합니다.

4. 느낀 점

(1) C/C++, Java, Python 프로그래밍 언어의 Readability, Writability, Reliability, Cost에 대해 느낀 점

- Readability

파이썬이 가장 가독성이 좋다고 생각됩니다. 하지만 이는 사람의 언어와 유사하다고 생각되어 이렇게 느껴지는 것이지, 과도한 줄인표현은 오히려 알아보기 힘들 수도 있습니다. C++과 Java는 객체지향언어로, 평소 자바로 문제해결을 해보지 않은 저에게는 낯설었습니다. 이처럼 문제해결에 주로 쓰는 언어에 따라 자바의 가독성은 다르게 느껴질 것 같습니다. C++은 저에게 있어 문제해결에 익숙한 언어라 한눈에 알아보기 좋다고 느껴졌습니다.

- Writability

제가 과제를 수행하는데 있어 프로그램을 짜기 제일 쉬운 언어는 C++이었습니다. 함수를 만드는데 있어 가장 원초적으로 접근할 수 있어 함수에 필요한 파라미터와 리턴값을 먼저 잡고 코드를 작성하면 되기 때문입니다. 그에 비해 파이썬에서의 함수는 생략한게 많아 오히려 고민을 하게 되었고, JAVA는 입출력방식이 낯설어 어렵게 느껴졌습니다.

- Reliability

신뢰성인데, 파일처리 수업에서 신뢰성은 고장나지 않는 것이라고 배웠습니다. 하지만 제가 느끼기에 과제를 수행하면서 신뢰성을 논할 만한 일이 없었기 때문에 세 언어 다 우수하다고 생각합니다.

- Cost

| 언어 | C++ | python | java |
|----|-------|-------------|----------|
| 시간 | 0.009 | 0.005081... | 0.010844 |

측정 시간은 모두 입력을 받고, 정렬하기 전을 기준으로 측정을 시작하였습니다. 왜 파이썬보다 C++이 더 시간이 크게 나왔는지에 대해서 의문이 듭니다. 보통 백준에서 알고리즘 문제를 풀 때에는 파이썬을 이용해서 문제를 풀었을 때가 시간초가가 나기 쉬웠는데, 왜 C++의 시간이 더 나왔는지에 대해 고민해보다, 제가 내린 해답은 컴파일러가 느리기때문이 아닐까라는 결론을 내렸습니다. 저는 현재 VS code를 사용해서 코딩 작업을 하는데, C/C++을 run 할 때마다 유독 속도가 느리다는 느낌을 받았습니다. VS code에서는 내장된 컴파일러가 아닌 외부 컴파일러를 쓰기 때문에 빌드도구에 따라 속도가 다를 수 있다는 해답을 얻게 되었습니다. 이번 과제를 계기로 빌드 도구를 바꿔보도록 하겠습니다.

(2) Chat GPT를 활용하여 자동으로 코드를 생성해 본 후, 본인이 스스로 작성한 코드와 비교하면서 느낀 점

단순히 C++로 짠 코드를 파이썬으로 변환해달라고 하니 이러한 답변이 나왔다.



죄송합니다. 해당 코드를 파이썬으로 변환하기 위해서는 몇 가지 추가적인 정보가 필요합니다. 예를 들어 `primeArr()` 함수에서 사용된 `vector` 클래스는 C++의 STL 라이브러리에 속한 클래스이므로, 이에 해당하는 파이썬 라이브러리가 무엇인지에 따라 코드를 다르게 작성해야 합니다. 또한, C++ 코드에서 사용된 `sort()` 함수는 파이썬의 리스트를 정렬하는 내장 함수인 `sorted()`로 대체할 수 있습니다. 따라서 이러한 정보들을 추가로 제공해주시면 보다 정확한 파이썬 코드로 변환해 드릴 수 있습니다.

하지만 문제를 직접 던져주었을 때에는 완벽한 코드가 나와서 어떻게 입력하느냐에 따라 답변이 좌지우지 된다는 것을 알게되었습니다. 그리고 다른

언어로 convert 하기 위해서는 최대한 상세한 언어에서 단순한 언어로의 변환이 가장 쉽다고 생각되었었는데, 위 답변처럼 C++의 STL이 다른 언어에서는 존재하지 않기 때문에 vector를 그대로 변환할 수 없다는 것을 알게 되어 옳지만은 않은 생각이라는 것을 알게 되었습니다.