DS HW12

시작일: 11월 27일 금요일

마감일: 12월 11일 금요일 23:59:59 까지 (여유있게 제출하세요!)

제출방법

```
submit pem_ta hw12_ ( "_" 부분에 1분반은 a, 2분반은 b)
```

과제 12는 정렬 알고리즘의 구현 및 성능 분석입니다. 책에 소개된 내부정렬 알고리즘들을 구현하고 임의의 리스트에 대한 실행시간을 측정하여 각 알고리즘의 복잡도를 분석하고 이해하는 것이 목표입니다.

구현할 정렬 알고리즘은 다음과 같습니다.

1. 삽입 정렬 (Insertion Sort)
2. 빠른 정렬 (Quick Sort)
3. 합병 정렬 (Merge Sort)
4. 순환적 합병 정렬 (Recursive Merge Sort)
5. 내츄럴 합병 정렬 (Natural Merge Sort)
6. 힙 정렬 (Heap Sort)

요구 사항

a. 제출 파일은 **Makefile, hw12.cpp, sort.h, 보고서(학번.tex, 학번.pdf)**입니다. **정확한 파일명은 중요합니다.**

b. 결과보고서는 과제의 이해도를 평가하는 중요한 척도입니다. 구현 과정과 분석결과를 **최대한 자세히** 작성해주세요

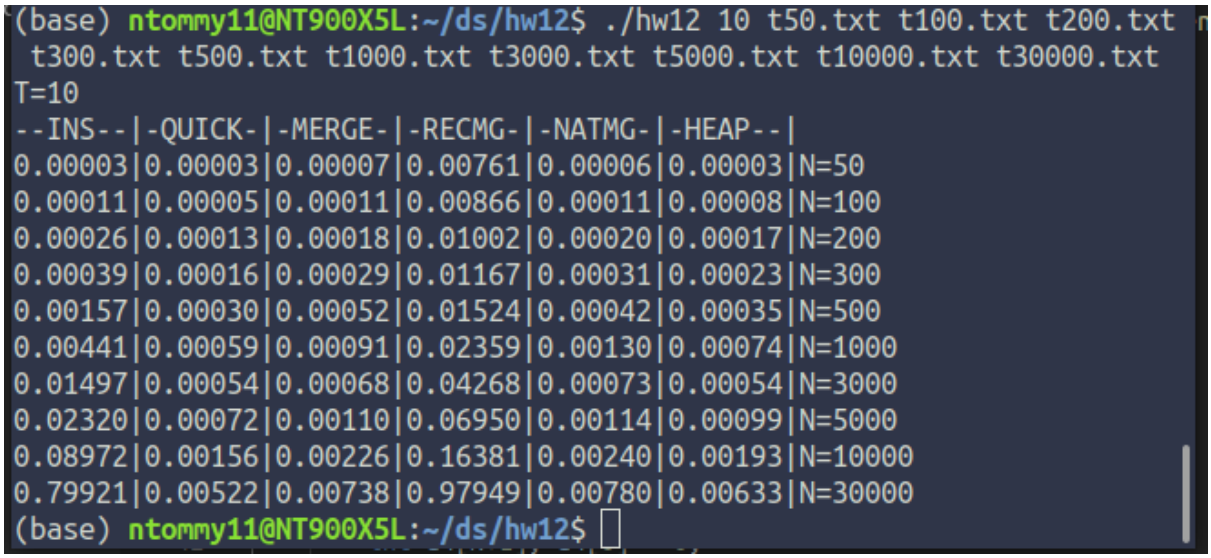
과제 설명

다음과 같이 작동하는 프로그램을 만들고자 합니다.

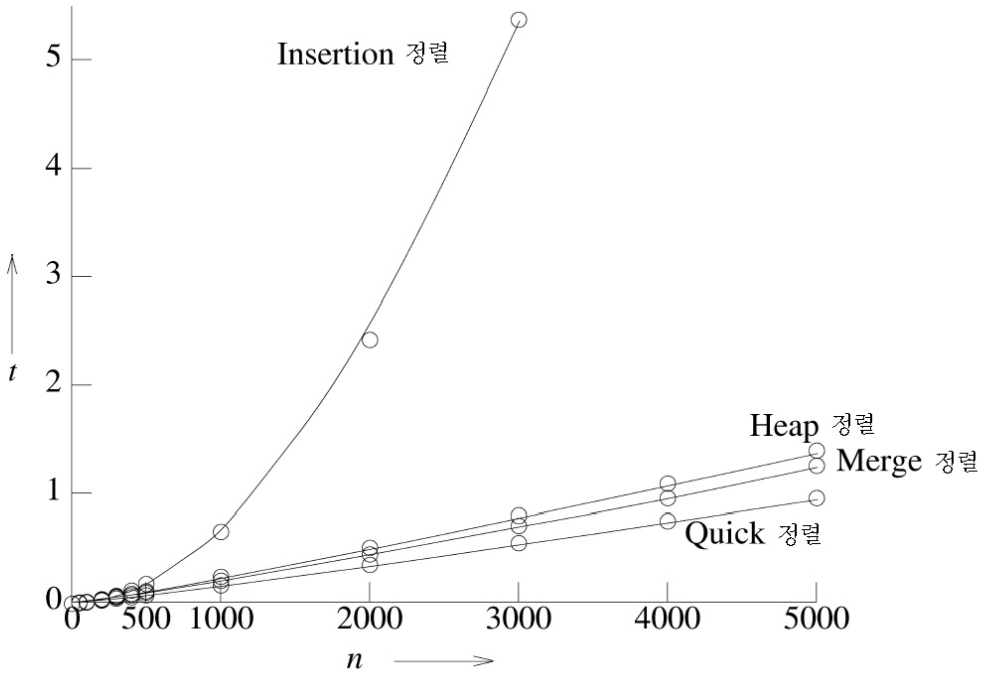
```
hw12 테스트케이스의 개수 파일명1 파일명2 파일명3 ...
```

- 실행파일 뒤의 첫번째 인자는 정렬 테스트케이스의 개수 T입니다.
- 그 뒤로 T개의 파일명이 나옵니다.

예시) T=10 의 경우 10개의 파일명이 뒤따라옵니다.



- 실행 시 위와같이 각 파일별로 실행시간을 측정하여 출력합니다. 단위는 초(s)입니다. (출력 부분은 이미 작성되어있습니다)
- 총 4개의 압축파일이 제공됩니다. 각 압축파일은 50~100000 사이의 데이터를 담은 리스트파일로 이루어져있습니다.
  - `random_t.zip` : 데이터가 랜덤으로 생성
  - `partially_sorted_t.zip` : 부분적으로 정렬된 리스트
  - `sorted_t.zip` : 완전 정렬된 리스트
  - `decreasing_t.zip` : 거꾸로 정렬된 리스트
- 각 압축파일은 15개의 파일로 이루어져있지만, 모든 파일을 다 사용할 필요는 없습니다. **단, 그래프를 그리기 위해 충분한 개수를 사용해야합니다**. 보고서에는 각 실행에 대해 위와 같이 캡쳐하여 첨부해야 합니다.
- 엑셀 또는 자신이 원하는 프로그램을 사용해서 아래와 같은 **그래프**를 그립니다. (강의노트를 참고). 직접 그려도 상관없습니다. 각 압축파일에 각각 적용합니다. 즉 4개의 캡쳐와 4개의 그래프를 생성합니다.



3. 코드

- Makefile

```
hw12: hw12.o
  g++ -o hw12 hw12.o
hw12.o: sort.h
```

- hw12.cpp

```
#include <iostream>
#include <fstream>
#include <ctime>

#include "sort.h"

using namespace std;

int main(int argc, char *argv[]){
    int T = atoi(argv[1]);  // num of test case
    cout << "T=" <<T<< endl;

    int N; // 각 테스트케이스 별 레코드의 길이
    int i; // iterator


    clock_t result[6]; // result 배열에 각 알고리즘 별로 실행시간을 저장하게 됩니다.
    // result[0]: insertion sort
    // result[1]: quick sort
```

```
    // result[2]: iterative merge sort
    // result[3]: reculsive merge sort
    // result[4]: natural merege sort
    // result[5]: heap sort
    clock_t start;
    clock_t end;


    if (argc < 3){
        cerr << "wrong argument count" << endl;
        return 1;
    }
    cout<<"--INS--|-QUICK-|-MERGE-|-RECMG-|-NATMG-|-HEAP--| "<< endl;

    for (i=2; i<T+2; i++){
        // i번째 인자의 파일을 읽습니다.
        // 각 정렬 알고리즘에 필요한 자료구조를 생성하고 데이터를 담습니다.

        // 여기부터 정렬 시간 측정을 시작합니다.

        /* example
        start = clock();
        삽입정렬 수행
        end = clock();
        result[0] = end-start; */


        // 결과를 출력합니다. (이 부분은 수정하지 않습니다)
        cout.precision(5);
        cout << fixed;
        for (j=0; j<6; j++){
            cout << (double)result[j]/CLOCKS_PER_SEC << "|";
        }
        cout << "N="<<N<<endl;
    }
}
```

- sort.h

```

```

결과보고서 내용

- 각 알고리즘별로 섹션을 만들어 작동방식을 설명하세요.
- 테스트 결과 (캡쳐 및 그래프)에 대해 분석하세요.

- 만약 부분적으로 구현못한 부분이 있다면, 어디서 막혔고 원인 등을 분석하여 작성하세요.

주의사항

- 파일명 정확히!
- 리눅스 서버 12시간 이상 오류가 없다면 기간 연장 없음.
- 메일로 제출 불가. 단, 마감시간 기준 4시간 전부터 제출이 불가능하다면 일단 메일로 보낸다.
- 남의 코드 배끼기 절대 금지 → F 주의. cheating 의심자는 webex로 면담 가능성이 있음
- 질문은 ntommy11@naver.com 으로 부탁합니다.