Tech University of Korea(TUK)
자료구조 과제 노트

202#-0#학기

담당교수	박정민
학번	•
이름	• 김##

한국공학대학교
컴퓨터공학부 (자료구조) 박정민 교수

과제수행 요청서

과제기간

■ (과제기간) 과제는 교재의 각 장이 끝난 일주일 후, 24시까지 제출

- **(제출방법)** e-class의 ▲과제게시판에 업로드
- **(제출파일명)** [학번이름].hwp **예시)** 01[2017131023박정민].hwp
- **(지각제출불가)** 과제를 제출하는 기간에 반드시 제출, ▲지각제출불가
- **(개인제출)** 과제제출은 ▲개인적으로 수행

과제내용

■ (과제내용) 과제의 내용은 총 4가지 1)요점정리, 2)역공부, 3)순공부, 4)자기성찰

- **(요점정리)** 수업시간의 내용을 ▲재정리 ▲ 개별적으로 공부한 것이 있다면 추가 요점정리
- **(역공부)** 코드를 분석한 내용정리 ▲손으로 분석한 그림, ▲디버깅 SW를 이용한 분석그림
- **(순공부)** 역공학을 통해 분석한 소스코드를 근거로 ▲코드를 수정/개선시켜보기, ▲주석 상세히 달기
- **(자기성찰)** ▲수업을 통해서 배운 것, ▲디버깅을 통해 집중적으로 공부한 것

결과물활용

■ (결과물) 결과물은 1)수시고사, 2)과제평가, 3) 개인 정리를 위한 지침

- **(수시고사를 위한 참조)** 과제들은 수시고사와 연계 ▲오픈북 시에 참조 문서로 활용
- **(과제평가)** 각 장별로 자료구조 수업이 종강될 때까지 작성하여 매주 평가
- **(개인정리를 위한 지침)** 과제내용 스스로 잘 정리하기 위한 좋은 가이드라인으로 활용

자료구조 과제 목차

1. 제1장 자료구조와 알고리즘

1-1 1장 자료구조와 알고리즘 요점정리

■ 자료구조

- 자료들을 정리하여 보관하는 여러 가지 구조
- 스택, 큐(줄을 선다), 그래프(그림을 그린다), 등 여러 형태가 존재

■ 알고리즘

- **컴퓨터로 문제를 풀기 위한 단계적인 절차**
- 알고리즘의 조건
  - 입력: 0개 이상의 입력이 존재하여야 한다.
  - 출력: 1개 이상의 출력이 존재하여야 한다.
  - 명백성: 각 명령어의 의미는 모호하지 않고 명확해야 한다.
  - 유한성 : 한정된 수의 단계 후에는 반드시 종료되어야 한다.
  - 유효성 : 각 명령어들은 실행 가능한 연산이여야 한다.
- 알고리즘의 기술 방법
  - 자연어: 인간이 읽기 쉬지만 약간의 모호성이 존재함
  - 순서도: 직관적이고 이해하기 쉽지만 복잡한 알고리즘의 경우 상당히 복잡해짐
  - 유사코드: 알고리즘 기술에 가장 많이 사용됨/ 알고리즘의 핵심적인 내용에만 집중가능
  - 프로그래밍 언어: 알고리즘의 가장 정확한 기술이 가능함/ 구체적인 사항들이 핵심내용 접근 방해함

■ 추상 자료형(ADT)

- 실제적인 구현으로부터 분리되어 정의된 자료형. 즉, **데이터 타임을 추상적으로 정의한 것**
- 데이터나 연산이 무엇(what)인가는 정의되지만,
  데이터나 연산이 어떻게(how) 컴퓨터 상에서 구현할 것인지는 정의되지 않는다.
- 추상화: 사용자에게 중요한 정보는 강조되고, 중요하지 않은 구현 세부 사항은 제거 하는 것

■ 알고리즘의 성능 분석

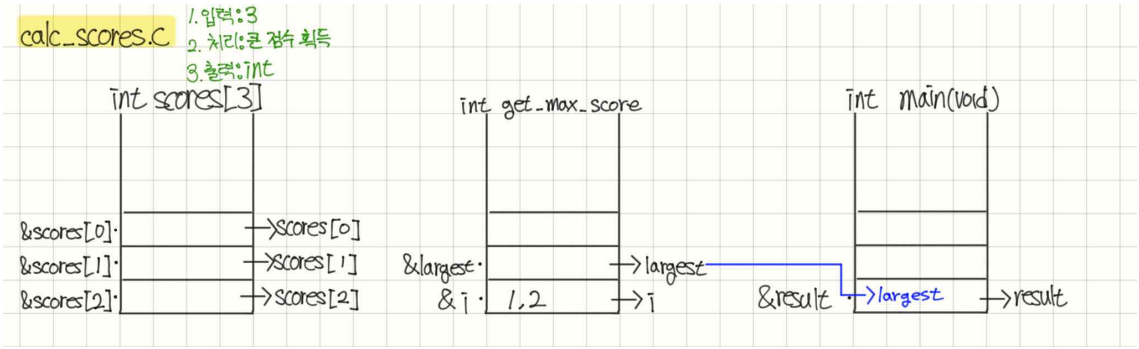
- 처리해야할 자료의 양이 많고, 사용자들은 빠른 프로그램을 선호→효율성 중요
- 부석 방법은 직접수행과 간접수행이 있다.
- 직접수행(수행 시간 측정)
  - 알고리즘을 프로그래밍 언어로 작성하여 실제 컴퓨터상에서 실행시킨 다음, 그 수행시간을 측정
  - 데이터의 개수가 같아야 함
  - 동일한 하드웨어를 사용하여야 함
- 간접수행(알고리즘의 복잡도 분석)
  - 직접 구현하지 않고서도 연산의 횟수를 측정하여 수행 시간을 분석하는 것
  - 일반적으로 연산의 횟수는 **n의 함수(=빅오 표기법)**
- 빅오 표기법(약 ~정도 값)
  - **연산의 횟수를 대략적(점근적)으로 표기한 것**
  - 자료의 개수가 많은 경우에는 가장 큰 항을 제외한 다른 항들은 상대적으로 무시될 수 있음

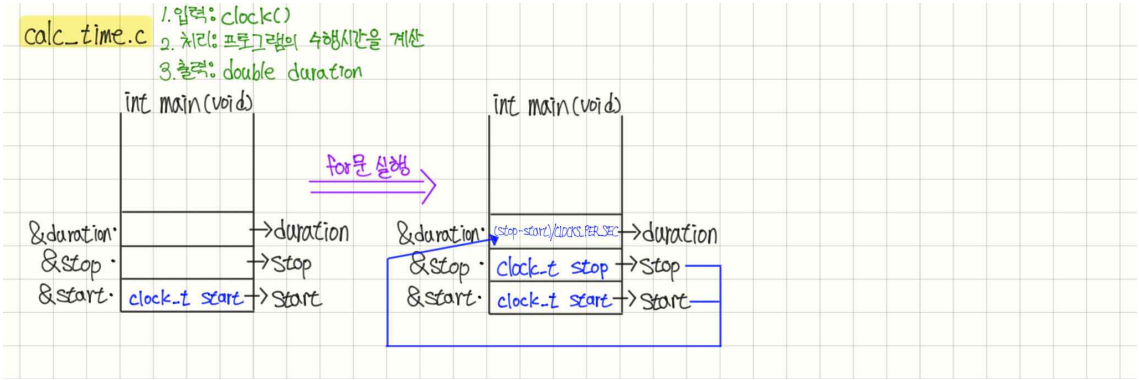
- 최선, 평균, 최악의 경우
  - 최선의 경우(best case): 수행 시간이 가장 빠른 경우/ 알고리즘에 따라서는 의미가 없어짐
  - 평균의 경우(average case): 수행 시간이 평균적인 경우/ 자료집합이 광범위해지면 구하기 힘듬
  - 최악의 경우(worst case): 수행 시간이 가장 늦은 경우

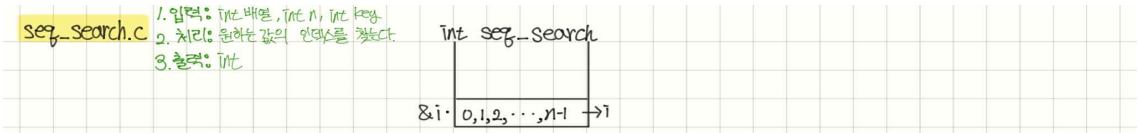
∴ **최악의 경우**의 수행시간이 알고리즘의 시간 복잡도 척도로 많이 쓰인다.

1-2 소스코드 디버깅 분석(역공부)

■ '손' 분석내용

- 01 calc_scores.c
  - 최고 점수를 구하는 알고리즘
  - 

- 02 calc_time.c
  - 프로그램의 수행시간을 직접 측정하는 방법
  - 

- 03 seq_search.c
  - 배열에서 원하는 값이 있는 인덱스를 찾아주는 알고리즘
  - 

■ 디버깅SW를 이용한 분석내용

- 01 calc_scores.c
  - scores를 전역배열로 선언 후 largest에 배열 첫 번째 인덱스의 값을 넣어준 다음 두 번째 인덱스부터 마지막 인덱스까지 값을 비교하여 if문을 사용해서 큰 값을 largest에 넣어준 뒤 반환한 값을 main 함수의 result값에 입력한다.
- 02 calc_time.c
  - time헤더 파일을 선언하고 정의되어 있는 함수를 호출해서 start변수와 stop변수를 각 연산의 시작과 끝에 위치시킨다. 그리고 start변수와 stop변수의 차를 CLOCKS_PER_SEC으로 나눈 값을 duration에 입력한 후 printf로 출력한다.
- 03 seq_search.c
  - for문을 사용하여 key값과 일치하는 인덱스를 찾을 때까지 반복한다. 탐색에 성공하면 인덱스값을 반환하고 실패하면 -1을 반환한다.

1-3 소스코드 수정/개선 & 주석(순공부)

■ 수정/개선 & 주석1 (수정/개선된 코드에 주석을 달고 & 설명)

- 01 calc_scores.c

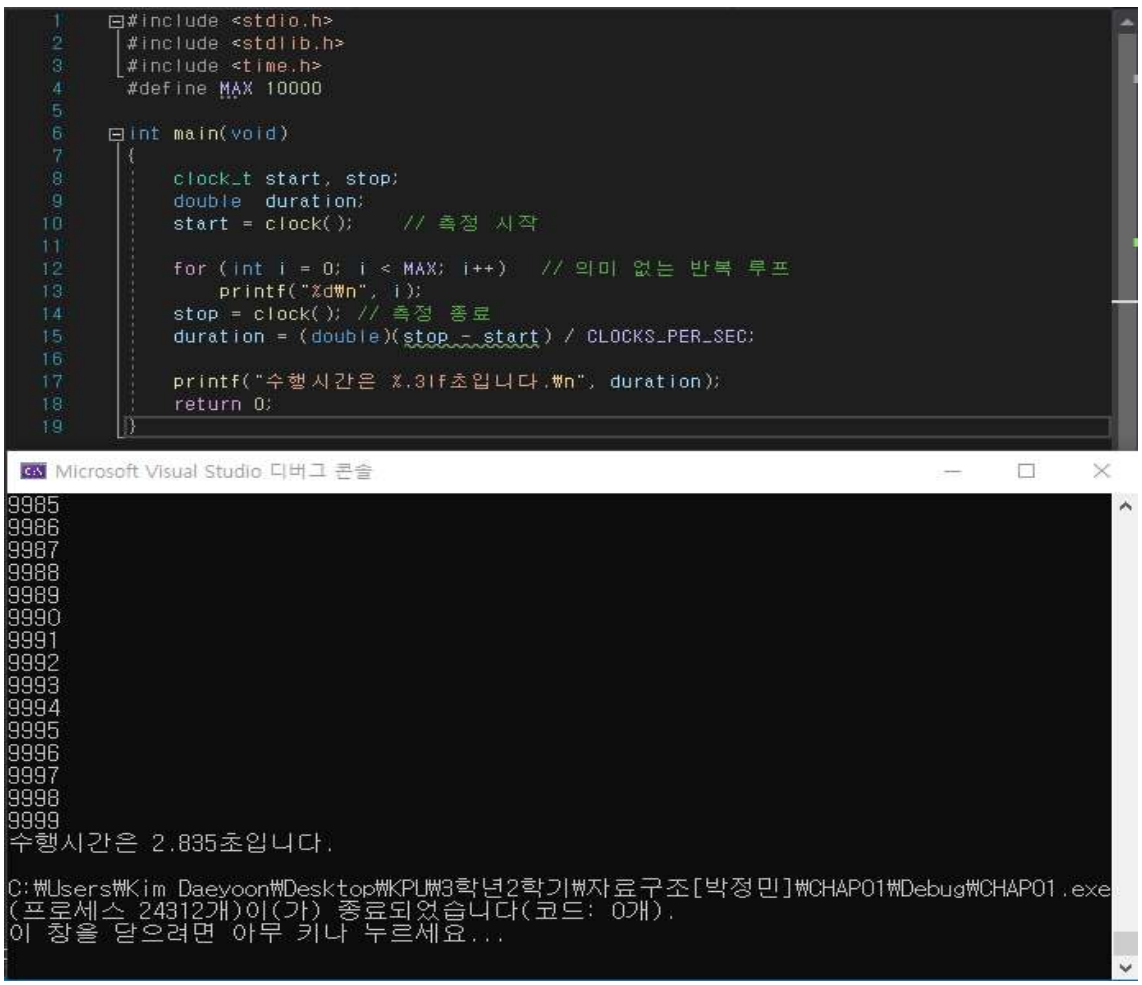
```
#include <stdio.h>
#include <stdlib.h>
#pragma warning(disable: 4996)    // scanf를 사용하기 위해 추가

#define MAX_ELEMENTS 3    // 3을 상수화
int scores[MAX_ELEMENTS];  // 자료구조, 전역배열

int get_max_score(int n);    // 함수의 선언, 함수의 리스트

int main(void) {      // 테스트
      int result;// 마지막에 출력할 값
      for (int i = 0; i < MAX_ELEMENTS; i++) {     // for, scanf를 사용하여 scores배열에 값을 입력
            scanf("%d", &scores[i]);
      }

      result = get_max_score(MAX_ELEMENTS);     // result에 get_max_score반환 값 largest를 입력
      printf("결과:%d\n ",result);      // 결과 출력

      return 0;
}

int get_max_score(int n) {     // 학생의 숫자는 n
      int i, largest;
      largest = scores[0]; // 알고리즘
      for (i = 1; i<n; i++) {
            if (scores[i] > largest) {
                  largest = scores[i];
            }
      }
      return largest;
}// 입력: n, 처리: 큰점수 획득, 출력: int(정수)
```

```c
#include <stdio.h>
#include <stdlib.h>
#pragma warning(disable: 4996)  // scanf를 사용하기 위해 추가

#define MAX_ELEMENTS 3  // 3을 상수화
int scores[MAX_ELEMENTS];   // 자료구조, 전역배열

int get_max_score(int n);   // 함수의 선언, 함수의 리스트

int main(void) {    // 테스트
    int result; // 마지막에 출력할 값
    for (int i = 0; i < MAX_ELEMENTS; i++) {    // for, scanf를 사용하여 scores배열에
        scanf("%d", &scores[i]);
    }

    result = get_max_score(MAX_ELEMENTS);   // result에 get_max_score반환 값 largest를
    printf("결과:%d₩n ", result);   // 결과 출력

    return 0;
}

int get_max_score(int n) {  // 학생의 숫자는 n
    int i, largest;
    largest = scores[0];    // 알고리즘
    for (i = 1; i < n; i++) {
        if (scores[i] > largest) {
            largest = scores[i];
        }
    }
    return largest;
}// 입력: n, 처리: 큰점수 획득, 출력: int(정수)
```

```
Microsoft Visual Studio 디버그 콘솔
15
33
19
결과:33

C:₩Users₩Kim Daeyoon₩Desktop₩KPU₩3학년2학기₩자료구조[박정민]₩CHAP01₩Debug₩CHAP01.exe
(프로세스 27440개)이(가) 종료되었습니다(코드: 0개).
이 창을 닫으려면 아무 키나 누르세요...
```

- 02 calc_time.c

```
#include <stdio.h>
#include <stdlib.h>
#include <time.h>   // clock, CLOCKS_PER_SEC 사용하기 위해 헤더파일 추가
#define MAX 10000 // 10000을 상수화

int main(void) // 테스트
{
	clock_t start, stop; // typedef long clock_t
	double  duration;   // 마지막에 출력할 값
	start = clock();// 측정 시작
	for (int i = 0; i < MAX; i++)   // 의미 없는 반복 루프
		printf("%d\n", i);     // printf를 MAX번 시행했을 때 소요되는 시간 측정
	stop = clock(); // 측정 종료
	duration = (double)(stop - start) / CLOCKS_PER_SEC; //   duration이 double형 변수이기 때문에
                                                             stop-start값을 double형으로 형 변환

	printf("수행시간은 %.3lf초입니다.\n", duration);//   duration의 값이 소수점 6자리까지 출력되는
	return 0                                              상황을 방지하기 위해 %.3lf로 서식문자 변경
}
```



- 03 seq_search.c

```
#include <stdio.h>
#include <time.h>   // 수행시간을 알기위해 헤더파일 추가
#pragma warning(disable: 4996)    // scanf를 사용하기 위해 추가
#define MAX 10     // 10 상수화

int main(void) // 테스트
{
      clock_t start, stop; //수행시간측정코드
      double  duration;
      int array1[MAX] = { 31,53,213,543,02,15,43,21,65,852 }; // 배열의 이름은 배열의 시작주소
      int result, search;  // 검색할 자료 search, 출력할 결과 result

      start = clock();// 측정 시작
      scanf("%d", &search);     // scanf를 이용하여 찾고 싶은 자료 입력
      result = seq_search(array1, MAX, search);   // result에 seq_search의 반환 값 i를 입력
      printf("%d\n", result);     // 결과 출력
      stop = clock(); // 측정 종료
      duration = (double)(stop - start) / CLOCKS_PER_SEC; //   duration이 double형 변수이기 때문에
                                                            stop-start값을 double형으로 형 변환
      printf("수행시간은 %.3lf초입니다.\n", duration);//   duration의 값이 소수점 6자리까지 출력되는
      return 0                                              상황을 방지하기 위해 %.3lf로 서식문자 변경
}

int seq_search(int *list, int n, int key)  // int* list = array; int n = 10; int key = 65
// 입력: 3개, 처리: 순차적으로 data를 찾는다, 출력 int 정수로 한다.
{
      int i;
      for (i = 0; i < n; i++)     // i는 0-9
      {
            if (list[i] == key)
                  return i;  /* 탐색에 성공하면 키 값의 인덱스 반환 */
      }
      return -1;     /* 탐색에 실패하면 -1 반환 */
}
```

```c
#include <stdio.h>
#include <time.h>
#pragma warning(disable: 4996)  // scanf를 사용하기 위해 추가

#define MAX 10

int main(void)
{
    clock_t start, stop;
    int array1[MAX] = { 31,53,213,543,02,15,43,21,65,852 }; // 배열의 이름은 배열의 시
    int result, search;
    double  duration;

    start = clock();    // 측정 시작
    scanf("%d", &search);
    result = seq_search(array1, MAX, search);
    printf("%d\n", result);
    stop = clock(); // 측정 종료
    duration = (double)(stop - start) / CLOCKS_PER_SEC;

    printf("수행시간은 %.3lf초입니다.\n", duration);

    return 0;
}

int seq_search(int *list, int n, int key)   // int* list = array; int n = 10; int key
// 입력: 3개, 처리: 순차적으로 data를 찾는다, 출력 int 정수로 한다.
{
    int i;
    for (i = 0; i < n; i++) // i는 0-9
    {
        if (list[i] == key)
            return i;  /* 탐색에 성공하면 키 값의 인덱스 반환 */
    }
    return -1;    /* 탐색에 실패하면 -1 반환 */
}
```

```
Microsoft Visual Studio 디버그 콘솔
543
3
수행시간은 0.649초입니다.

C:\Users\Kim Daeyoon\Desktop\KPU\3학년2학기\자료구조[박정민]\CHAP01\Debug\CHAP01.exe
(프로세스 21564개)이(가) 종료되었습니다(코드: 0개).
이 창을 닫으려면 아무 키나 누르세요...
```

2. 자기성찰

2-1 평가내용 및 느낀 점 (총 50점)

평가 내용 (30점 만점)	3점	2점	1점
1. 온라인강의는 주어진 기한 내 듣기 완료를 하였나?	●		
2. 이번 장의 수업은 이해도가 높았나?	●		
3. 역공부, 주어진 소스코드 분석을 위해 손으로 그림을 그려보았나?	●		
4. 역공부, 주어진 소스코드 분석을 위해 디버깅 SW로 분석해보았나?	●		
5. 역공부, 소스코드를 구현한 개발자의 노력을 생각해보았나?	●		
6. 순공부, 3번과 4번을 수행하면서 코드를 개선시킬 노력은 하였나?	●		
7. 순공부, 개선코드 주석은 꼼꼼히 작성하였나?	●		
8. 과제 '양식(hwp)'을 준수하여 작성하였나?	●		
9. 오타 없이 표의 양식, 그림의 넓이&높이 등 정돈된 보고서인가?	●		
10. 시간 마감에 급급한 대충 작성한 보고서가 아닌, 열심히 공부한 정직한 보고서인가?	●		
총 점	30 점		

'객관식 30점, 주관식 20점, 총 50점을 스스로 자기성찰' 하고, 정직한 보고서를 위해 최선을 다하자!

노력지수 (10점 만점)	• 9점 - 여름방학기간동안 자료구조과목을 예습했던 것 덕분에 이해하기가 쉬웠고, 예습할 때에는 이해가 잘 안되던 부분까지 이번 수업을 통해 이해를 할 수 있었다. 또한 이번에 역공부로 하면서 그렸던 덕분에 프로그램이 돌아가는 구도를 좀 더 확실히 알게 되었다.
느낀 점 (10점 만점)	• 10점 - 노력지수에서 언급한 것처럼 예습할 때에는 이해가 잘 안되던 부분까지 이해하게 되어 굉장히 시원한 기분이었다. 그리고 평소에는 한 번도 해보지 않던 공부법을 이용해봤는데 비록 처음에는 어색하고 서툴러서 시간이 많이 걸렸지만, 과제를 완료하니까 머리에 남는 정보의 질이 다른 느낌이여서 노력할 가치가 있다는 생각이 들었다.

- '자기성찰' 부분은 과제 채점 시, 많은 부분 참고가 됩니다. -