실습 1 – 선택 정렬과 이진 검색

- 실습의 목표
  - 선택 정렬과 이진 검색 알고리즘의 동작 과정을 이해한다.
  - 입력 데이터의 크기 변화에 따른 실행 시간의 변화를 이해한다.

선택정렬의 실습

- 프로그램 1.4의 선택정렬 void sort() 구현
  - 함수원형 : void sort(int list[], int n)

- **(1.1.)** main()함수에서 무작위로 100,000개의 정수를 생성한 다음, int A[100000] 배열에 저장한 후, sort 함수 호출
  - sort() 함수 호출한 후, A의 내용이 오름차순으로 정렬되었는지를 확인하는 is_sorted(A) 함수 호출
    - 함수원형 : int is_sorted(int A[], int n) 정렬되었는지 확인
    - 무작위 정수의 범위는 1 ~ 100000
    - is_sorted: 모든 i에 대해 A[i] <= A[i+1]인지를 확인하여 참일 경우, 1을 return. 아니면 0을 return

선택정렬의 실습

- **(1.2.)** n의 크기를 100부터 1,000,000까지 바꾸면서 sort 함수의 실행 시간을 측정하여 관찰지에 기록
  - 역순 정렬된 배열 활용 최악 상황일때 : 역순 정렬 //1,000,000
  - 입력 정수 범위는 1 ~ 배열크기
  - 배열의 크기를 최대한 크게 할 수 있는 방법 모색
  - 실행 시간 측정 방법: 프로그램 1.24 참조
  - n과 실행 시간을 Excel에 저장한 후, 그래프를 그려볼 것!

1.24 clock

이진 검색의 실습

- 선택 정렬의 검색 방법과 거의 동일
- 반복문을 이용한 이진 검색 알고리즘(프로그램 1.7) 구현
  - 함수원형 :
    int binsearch(int list[], int searchnum, int left, int right)
- **(1.3.)** main() 함수에서 100,000개의 정수를 무작위로 생성하여, A 배열에 저장한 후 sort() 함수를 호출하여 오름차순으로 정렬
  - 무작위 정수의 범위는 1 ~ 100000
- 이후 binsearch(A, 검색 데이터) 함수를 호출하여 return된 곳에 검색 데이터가 존재하는지 확인
  - binsearch 함수에서 int 변수 counter를 0으로 초기화하고, while문을 돌 때마다 1씩 증가. return할 때, counter의 값이 얼마인지를 출력

이진 검색의 실습

- **(1.4.)** 배열의 크기를 변화시키면서 counter의 값을 관찰
  - 역순 정렬된 배열 활용
  - 입력 정수 범위는 1 ~ 배열크기
  - 50,000 및 100,000의 배열 크기에서 확인
  - sort() 함수를 호출하여 오름차순으로 정렬
  - is_sorted() 함수 호출하여 정렬되었는지 확인
  - 검색 데이터의 counter 값을 관찰 및 관찰지에 기록