알고리즘 과제

Practice.05

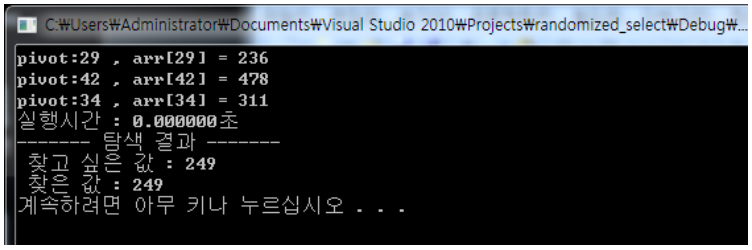
학번 : 201402432

이름 : 조디모데

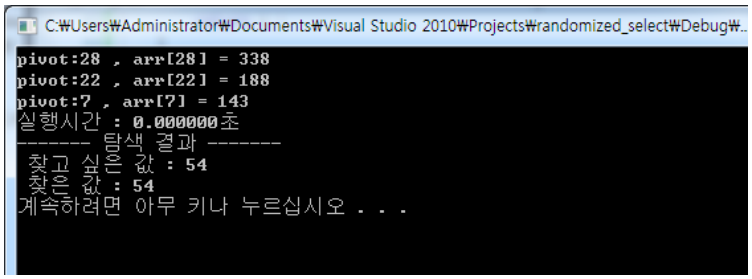
Randomized_Select.

- Input_50

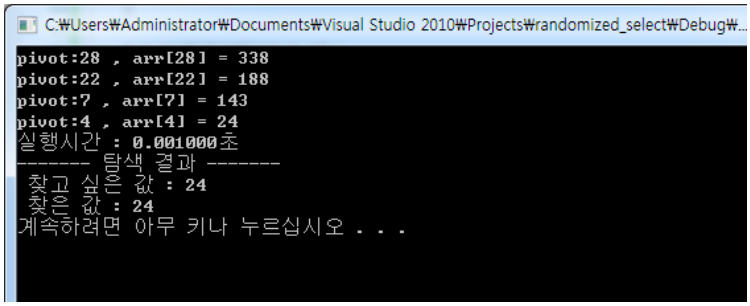
  - 10th



  - 25th



  - 30th



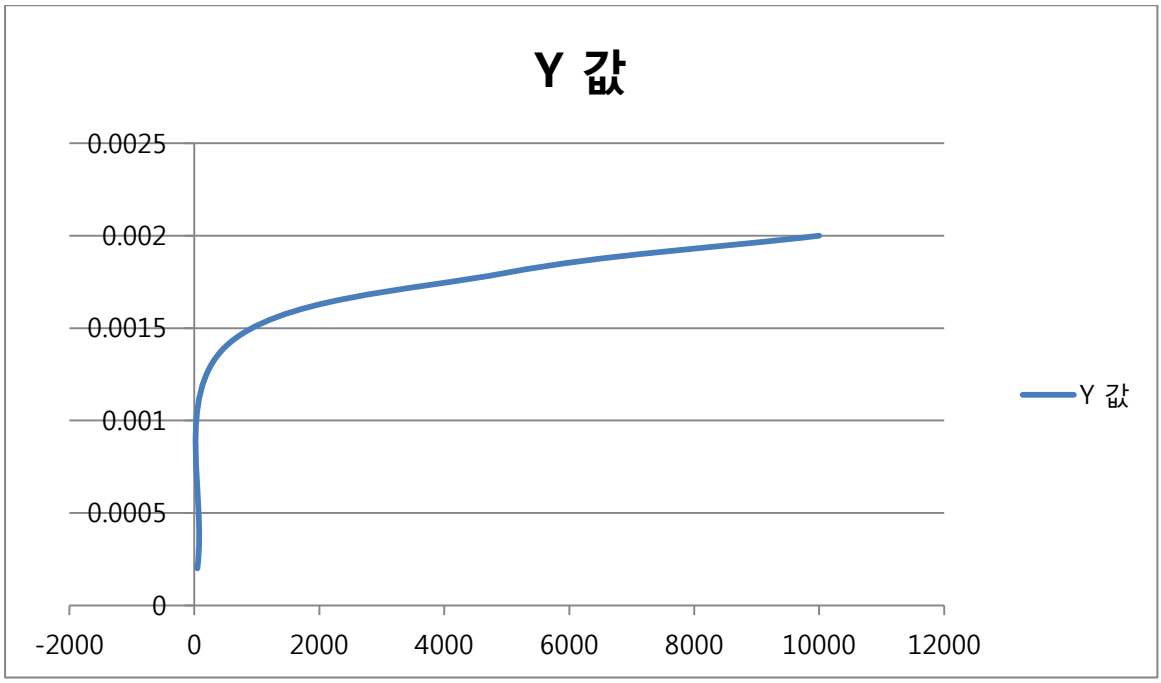
- 알고리즘 설명

  퀵 정렬과 유사하게 피봇을 정하여 배열을 두개로 나누어 가면서 찾고자 하는 값과 피봇을 비교하여 탐색 범위를 좁혀가는 알고리즘
  이때 피봇은 랜덤으로 정해진다.

- 컴파일 방법

  input 폴더를 다음 위치에 넣는다
  C:₩₩Users₩₩Administrator₩₩Desktop₩₩input₩₩input50.txt
  찾고자 하는 값을 설정find = num[n] (code:76)
  input의 크기에 맞게 num, size 변수 초기화

● 수행시간을 그래프



Input

50 : 0.0002

500 : 0.0014

5000 : 0.0018

10000 : 0.0020

● Code (.C)

```
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
#include <memory.h>

void swap(int *arr, int i, int j){
   int tmp ;
   if(i != j){
         tmp = arr[i] ;
         arr[i] = arr[j] ;
         arr[j] = tmp ;
   }
}
void printArray(int* arr,int length)
{
   int i ;
   for(i=0 ; i<length ; i++){
      printf("INDEX : %d, VALUE : %d₩n",i,arr[i]) ;
   }
}
int partition(int *arr, int p, int r){
   int i, j, tmp ;
   int pivot = p+rand()%(r-p) ;
   swap(arr, p, pivot) ;
   i = p+1 ;
   for(j=p+1 ; j<=r ; j++){
      if(arr[j]<=arr[p]){
         swap(arr, j, i) ;
         i++ ;
      }
   }
   swap(arr, pivot, i-1) ;
   return i-1 ;
}

int rand_select(int *arr, int p, int r, int i) { // 배öe열¯¡©, 시öA작U, 찾¡Ì는¥A 값Æ¨£,
끝øǿ®
        int q, k ;
   if(arr[p]==r){
           return p ;
   }

   if(p==i){
           return i ;
   }

   q = partition(arr,p,i) ;
   printf("pivot:%d , arr[%d] = %d₩n",q, q,arr[q]) ;

  if(arr[q]==r){
          return q ;
  }
```

```
    else if(r<arr[q])
        return rand_select(arr, p, r, q-1) ;
    else
        return rand_select(arr, q+1, r,i) ;
}



int main(void) {
    int i,size , maxValue, num[51], find ;
    FILE *fps ;
    int temp = 0 ;
    int result ;
    clock_t start,end ;

    size = 50 ;
    // input의C 숫ùy자U들ie을 ¡ í 배öe열¯ ¡© num에¯¢® 저u장a하I는¥A 부¬I분¬¨¢
    fps = fopen("C:₩₩Users₩₩Administrator₩₩Desktop₩₩input₩₩input50.txt","rt") ;
    for(i=0; i <= size ; i++){
        fscanf(fps,"%d",&temp) ;
        num[i]=temp ;
    }
    fclose(fps) ;

    find = num[25] ;

    // 정¢´렬¤A 시öA작U 시öA간Æ ¡Ì 저u장a
    start = clock() ;

    // 정¢´렬¤A하I는¥A 부¬I분¬¨¢
    result = rand_select(num, 0, find, size-1) ;
    // 정¢´렬¤A 후A 시öA간Æ ¡Ì 저u장a
    end = clock() ;

    // 실öC행a 시öA간Æ ¡Ì 출a력¤A
    printf("실öC행a시öA간Æ ¡Ì : %lf초E₩n",(end-start)/(double)1000) ;

    // 정¢´렬¤A 결Æa과Æu 출a력¤A
    printf("------- 탐¨öi색îo 결Æa과Æu ------- ₩n 찾 ¡Ìi고Æi 싶öI은¨¬ 값Æ¨£ : %d₩n
찾 ¡Ì은¨¬ 값Æ¨£ : %d₩n",find,num[result]) ;
    //printArray(result, size) ;

    system("pause") ;

    return 0;
}
```

Selection in Worst-Case Linear Time Algorithm.

- 알고리즘 설명

  5.1 에서 랜덤으로 피봇을 선택 했다면, 5.2 과제에서는 피봇을 선정하는 방식이 다르다.
  1. 전체 배열을 원소 5 개씩 나누고 나머지가 있다면 추가로 배열을 만들어 추가한다.
  2. 5 개의 원소를 정렬하여 가운데 값을 찾고 찾은 값들 중 다시 중간 값을 찾는다.
  3. 찾은 중간의 중간 값을 피봇으로 select 를 진행한다.

- 컴파일 방법

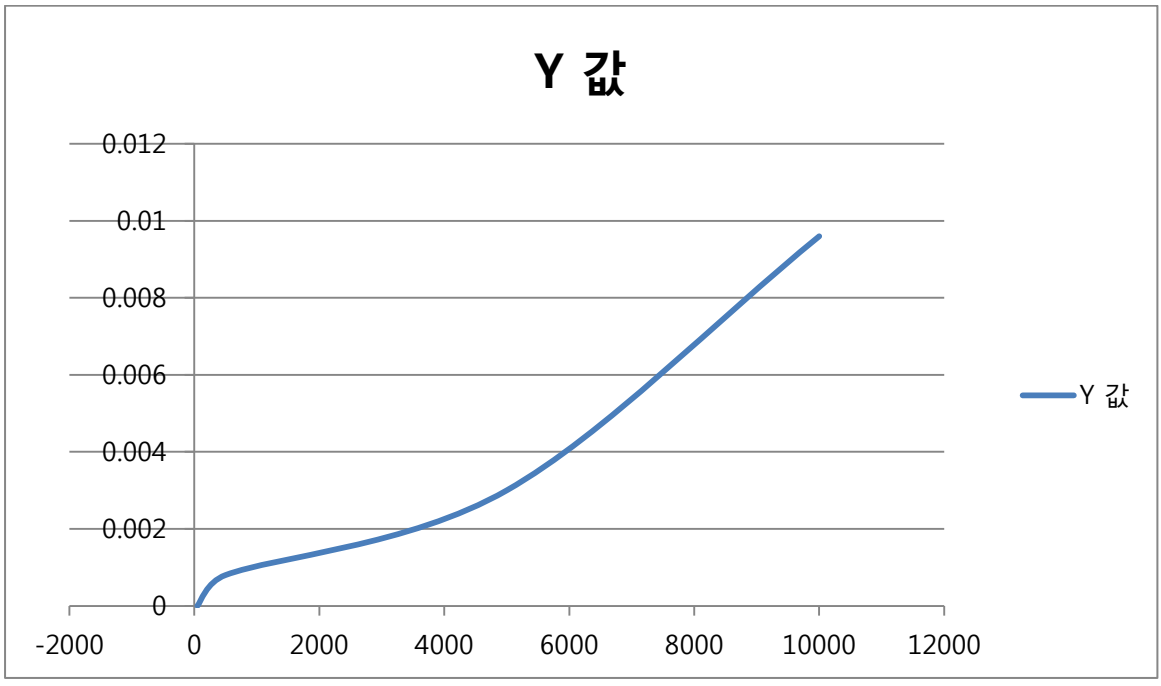
  input 폴더를 다음 위치에 넣는다

  C:₩₩Users₩₩Administrator₩₩Desktop₩₩input₩₩input50.txt

  찾고자 하는 값을 설정find = num[n] (code:76)

  input의 크기에 맞게 num, size 변수 초기화

- 수행시간을 그래프



Input

50 : 0.0000

500 : 0.0008

5000 : 0.0030

10000 : 0.0096

● Code (.C)

```
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
#include <memory.h>

void swap(int *arr, int i, int j){
   int tmp ;
   if(i != j && j>=0 && i>=0){
        tmp = arr[i] ;
        arr[i] = arr[j] ;
        arr[j] = tmp ;
   }
}

int findMid(int *arr, int i, int j){
        int k, q ;
        for(q=i ; q<j ; q++){
                for(k=q ; k<j ; k++){
                        if(arr[i]>arr[k])
                                swap(arr, i, k) ;
                }
        }
        return i+(j-i)/2 ;
}
int getPivot(int *arr, int i, int j){
        int k, q, *mid, index ;
        mid = (int*)malloc(sizeof(int)*((j-i)/5+1) ) ;
        memset(mid,0,(j-i)/5+1) ;
        index = 0 ;
        for(k=i ; k<j ; k=k+5){
                mid[index]=findMid(arr,i,i+5) ;
                index ++ ;
                if(k+5>j){
                        mid[index] = findMid(arr,k,j) ;
                        index++ ;
                        break ;
                }
        }

        for(k=0 ; k<index ; k++){
                for(q=k ; q<index ; q++){
                        if(mid[k]>mid[q])
                                swap(mid, k, q) ;
                }
        }

        return mid[index/2] ;

}

void printArray(int* arr,int length)
{
```

```
   int i ;
   for(i=0 ; i<length ; i++){
      printf("INDEX : %d, VALUE : %d₩n",i,arr[i]) ;
   }
}
int partition(int *arr, int p, int r){
   int i, j, tmp ;
   int pivot = getPivot(arr, p, r) ;
   swap(arr, p, pivot) ;
   i = p+1 ;
   for(j=p+1 ; j<=r ; j++){
      if(arr[j]<=arr[p]){
         swap(arr, j, i) ;
         i++ ;
      }
   }
   swap(arr, pivot, i-1) ;
   return i ;
}

int rand_select(int *arr, int p, int r, int i) { // 배öe열¯¡©, 시öA작U, 찾¡Ì는¥A 값Æ¨£,
끝øǿ®
        int q, k ;
   if(arr[p]==r){
           return arr[p] ;
   }

   if(p==i){
           return arr[i] ;
   }

   q = partition(arr,p,i) ;
   printf("pivot:%d , arr[%d] = %d₩n",q, q,arr[q]) ;

  if(arr[q]==r){
          return arr[q] ;
  }
   else if(r<arr[q])
      return rand_select(arr, p, r, q-1) ;
   else
      return rand_select(arr, q+1, r,i) ;
}



int main(void) {
   int i,size , maxValue, num[51], find ;
   FILE *fps ;
   int temp = 0 ;
   int result ;
   clock_t start,end ;

   size = 50 ;
   // input의C 숫ùy자U들ie을 ¡í 배öe열¯¡© num에¯ǿ® 저u장a하I는¥A 부¬I분¬¨ǿ
```

```
    fps = fopen("C:₩₩Users₩₩Administrator₩₩Desktop₩₩input₩₩input50.txt","rt") ;
    for(i=0; i < size ; i++){
       fscanf(fps,"%d",&temp) ;
       num[i]=temp ;
    }
    fclose(fps) ;

    find = num[30] ;

    // 정¢´렬¤A 시öA작U 시öA간Æ¡Ì 저u장a
    start = clock() ;

    // 정¢´렬¤A하I는¥A 부¬I분¬¨¢
    temp = rand_select(num, 0, find, size-1) ;
    // 정¢´렬¤A 후A 시öA간Æ¡Ì 저u장a
    end = clock() ;

    // 실öC행a 시öA간Æ¡Ì 출a력¤A
    printf("실öC행a시öA간Æ¡Ì : %lf초E₩n",(end-start)/(double)1000) ;

    // 정¢´렬¤A 결Æa과Æu 출a력¤A
    printf("------- 정¢´렬¤A 결Æa과Æu ------- ₩n%d == %d₩n",find,temp) ;
    //printArray(result, size) ;

    system("pause") ;

    return 0;
}
```