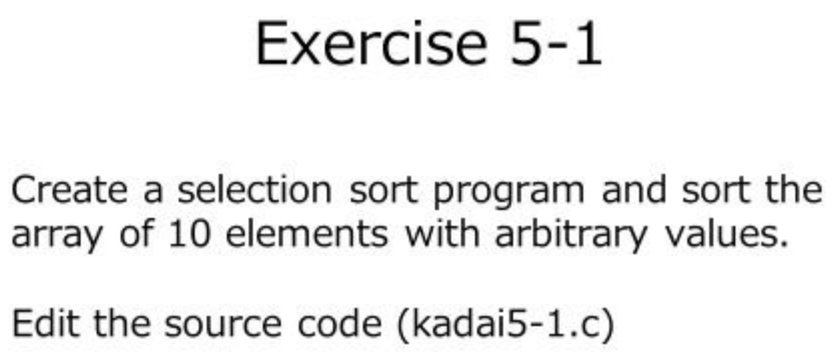
2020 College of Engineering Systems
Introduction to programming B 【exercise-week05】

Student ID（202018021） Name（Fu, Ziyu）

Answer the following questions. You can change the answer space freely.

【Exercise 5-1】



Exercise 5-1

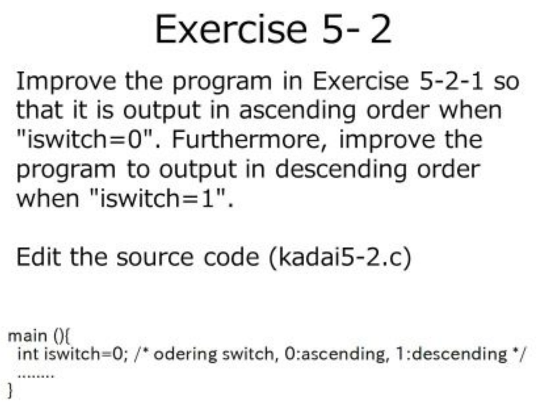
Create a selection sort program and sort the array of 10 elements with arbitrary values.

Edit the source code (kadai5-1.c)

(See next page)

```c
#include<stdio.h>

int num_list[10]= {2,5,7,9,3,1,0,6,4,8};
int selection_sort(int arr[], int len);

int main(){
    int i;

    selection_sort(num_list, 10);

    for(i=0;i<10;i++){
        printf("%d ",num_list[i]);
    }
    printf("\n");

}

int selection_sort(int arr[], int len){
    int i, j, min, temp;
    for (i = 0; i < len-1; ++i){
        min = i;
        for (j = i+1; j < len; ++j){
            if (arr[j] < arr[min]){
                min = j;
            }
        }

        temp = arr[min];
        arr[min] = arr[i];
        arr[i] = temp;
    }

    return 0;
}
```

【Exercise 5-2】



Exercise 5- 2

Improve the program in Exercise 5-2-1 so that it is output in ascending order when "iswitch=0". Furthermore, improve the program to output in descending order when "iswitch=1".

Edit the source code (kadai5-2.c)

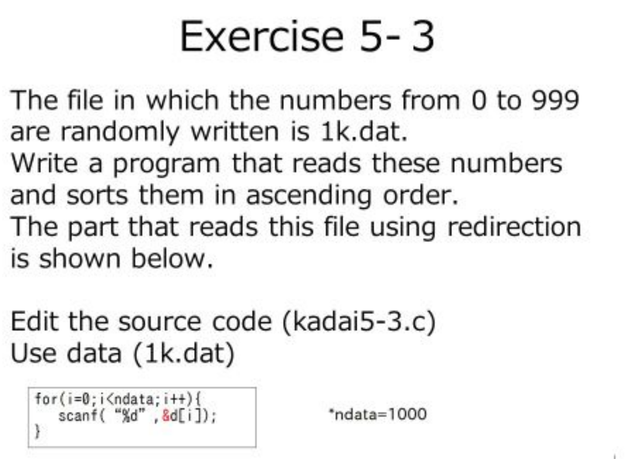
```
main (){
 int iswitch=0; /* odering switch, 0:ascending, 1:descending */
 ........
}
```

```
#include<stdio.h>


int num_list[10]= {2,5,7,9,3,1,0,6,4,8};

void selection_sort();


int main(){


    int iswitch = 1;/*odering switch, 0:ascendeing, 1:descending*/


    int i;


    selection_sort(iswitch);


    for(i=0;i<10;i++){

        printf("%d",num_list[i]);

    }

    printf("\n");


}


void selection_sort(int mode){

    int i, j, min, temp;

    for (i = 0; i < 10-1; ++i){

        min = i;

        for (j = i+1; j < 10; ++j){
            if (num_list[j] < num_list[min]){
                min = j;
            }
        }
        temp = num_list[min];
        num_list[min] = num_list[i];
        num_list[i] = temp;
    }

    if (mode == 1){
        int start = 0, end = 9;
        while (start < end){
        int temp = num_list[start];
        num_list[start] = num_list[end];
        num_list[end] = temp;
        start++;
        end--;
        }
    }


}
```

NOTE:

I'm not sure if this is what you were looking for. The problem might have wanted another selection sort in "reverse logic". That wouldn't be hard to make either though. The algorithm would just need to look for the largest element in the array and move it over to the right most. I find it easier to flip the array entirely by the end if the mode flag is 1. Admittedly, this adds another O(N) to the time complexity of the algorithm, but this shouldn't matter because the average time complexity of selection sort is O(N^2).

【Exercise 5-3】



Exercise 5- 3

The file in which the numbers from 0 to 999 are randomly written is 1k.dat.
Write a program that reads these numbers and sorts them in ascending order.
The part that reads this file using redirection is shown below.

Edit the source code (kadai5-3.c)
Use data (1k.dat)

```
for(i=0;i<ndata;i++){
    scanf( "%d" ,&d[i]);
}
```

*ndata=1000

```
#include<stdio.h>


//int num_list[10]= {2,5,7,9,3,1,0,6,4,8};
int selection_sort();


int main(){
    int i;
    int ndata = 1000;
    int d[ndata];


    for(i=0;i<ndata;i++){
        scanf("%d",&d[i]);
    }


    selection_sort(d,ndata);


    for(i=0;i<ndata;i++){
        printf("%d\n",d[i]);
    }
    printf("\n");


}
```

```c
int selection_sort(int num_list[],int ndata){


    int i, j, min, temp;

    for (i = 0; i < ndata-1; ++i){
        min = i;
        for (j = i+1; j < ndata; ++j){
            if (num_list[j] < num_list[min]){
                min = j;
            }
        }

        temp = num_list[min];
        num_list[min] = num_list[i];
        num_list[i] = temp;
    }



    return 0;
}
```