```
#include <stdio.h>
#include "utils.h"
void input_array(int *a, int n) {                    *per l'Item invece di int *a metti Item a[ ]
        int i;
        for(i=0; i<n ; i++)
                scanf("%d", &a[i]);                   a[i]=inputItem();
}
-------------------------------------------------------------------------------------------------------------
void output_array(int *a, int n) {                    *per l'Item invece di int *a metti Item a[ ]
        int i;
        for(i=0; i<n ; i++)
                printf("%d ", a[i]);                  outputItem(a[i])
}
-------------------------------------------------------------------------------------------------------------
int min( int *a, int n) {
        int min=0, i;
        for(i=1; i<n ; i++)
                if(a[i] <a[min])
                        min=i;

        return min;
}
-------------------------------------------------------------------------------------------------------------
void selection_sort(int *a, int n) {
        int i;
        int m;
        for(i=0;i<n-1;i++) {
                m = min(a+i,n-i)+i;
                swap(&a[i],&a[m]);
        }
```

```
void insertSortedArray(int *array, int *size, int element) {

	int i;
	for(i=*size;i>0;i--)
		if(element < array[i-1])
			array[i] = array[i-1];
		else
			break;

	array[i] = element;
	*size += 1;
	//(*size)++;
}
```

```
void insertion_sort(int *array, int size) {

	int i;
	for(i=1;i<size;)
		insertSortedArray(array,&i,array[i]);
}
```

```
void bubbleSort(int *array, int n) {
	int i,j;
	for(i=1;i<n;i++) //nella prima iterata scambiamo elementi adiacenti fino alla fine
		for(j=0;j<n-i;j++) //faremo un'iterazione in meno, perchè non arriviamo all'ultimo elemento
		ma sempre al penultimo
			if(array[j]>array[j+1]) //perchè scambiamo elementi adiacenti
				swap(&array[j],&array[j+1]);

}
```

```
int adaptiveBubbleSort(int *array, int n) {
	int i,j, ordinato=0, contatore=0; // ci contiamo quanti scambi fa;
	for(i=1;i<n && !ordinato;i++) { //nella prima iterata scambiamo elementi adiacenti fino alla fine
		ordinato=1;
		for(j=0;j<n-i;j++){ //faremo un'iterazione in meno, perchè non arriviamo all'ultimo elemento ma sempre al penultimo
			contatore++;
			if(array[j]>array[j+1]) { //perchè scambiamo elementi adiacenti
				ordinato=0;
				swap(&array[j],&array[j+1]);
			}
		}
	}
	return contatore;
}
```

```
int ricercaBin(int *arr, int n, int element) {
	int inizio=0,centro, fine= n-1;

	while(fine>=inizio) {
		centro=(inizio+fine)/2;
		if(element == arr[centro])
			return centro;
		else if(element < arr[centro])
			fine=centro-1;
		else if(element > arr[centro])
			inizio=centro+1;
	}
	return -1; //Se usciamo dal while restituiamo -1 perchè non abbiamo trovato l'elemento
}
```

```
int ricercaLineare(int *arr, int n, int el) {
        int i=0;

        while(i<n && a[i] <el)
                i++;
        return (i<n && a[i] == el ? i: -1);

}
```

```
void delete_element( int a[], int *n, int pos) {
        int i;
        for(i=pos;i<*n-1;i++)
        a[i] = a[i+1];
        --n*;
}
```

```
void insert_array(int a[+, int *n, int el, int pos) {
        Int i;
        if(pos<0 !! pos>*n)
        return;
        for(i=*n; i> pos; i--)
                a[i]= a[i-1]
        a[pos] = el;
        ++*n;
```

```
int search_element(int a[], int n, int el) {
        int i;
        for(i=0; i<n ; i++)
        if(a[i] == el)
         return i;
        return -1;
}
```

SELECTION SORT RICORSIVO SU LISTA

```
Struct node * minimo(struct node *p) {
 Struct node *i,*min=p;
for(i=p;i!=null;i=i->next) {
if(cmpItem(min->item) i->item))>0)
min=i;
}
Return min;
}

Void selectionSortRicorsivo(struct node *p) {
If(p!=null) {
Struct node *pos_minimo = minimo(p);
swap(&(pos_minimo->item),& (p->item));
selectionSortRicorsiva(p->next);
}
}

Void selectionSort(List list) {
selectionSortRicorsivo(list->head);
}
```

SELECTION SORT RICORSIVO SU INTERI O STRINGHE

```
Void selectionSort(Item *a[], int n) {
Int i =0,min;
for(i=0;i<n-1;i++)
        min = minimo(a+i,n-i);
        swap(&a[i],&a[min+i]);
}

Void selectionSortRicorsivo(Item a[],int n) {
If(n==1)
Return;
int min = minimo(a+1,n-1)+1;
if(cmpItem(a[min],a[0]<0)
        swap(&a[0],&a[min]);
selectionSortRicorsivo(a+1,n-1);
}

int min( Item *a, int n) {
        int min=0, i;
        for(i=1; i<n ; i++)
                if(cmpItem(a[i],a[min]<0)
                        min=i;

        return min;
}
```