```
#include <stdio.h>
#include <math.h>
#define MAX_SIZE 101
#define SWAP(x,y,t) ((t) = (x), (x)= (y), (y) = (t))
void sort(int [],int); /*selection sort */
void main(void)
{
   int i,n;
   int list[MAX_SIZE];
   printf("Enter the number of numbers to generate: ");
   scanf("%d",&n);
   if( n < 1 || n > MAX_SIZE) {
     fprintf(stderr, "Improper value of n\n");
     exit(EXIT_FAILURE);
   }
   for (i = 0; i < n; i++) {/*randomly generate numbers*/
      list[i] = rand() % 1000;
      printf("%d  ",list[i]);
   }
   sort(list,n);
   printf("\n Sorted array:\n ");
   for (i = 0; i < n; i++) /* print out sorted numbers */
      printf("%d  ",list[i]);
   printf("\n");
}
void sort(int list[],int n)
{
   int i, j, min, temp;
   for (i = 0; i < n-1; i++)  {
      min = i;
      for (j = i+1; j < n; j++)
         if (list[j] < list[min])
            min = j;
      SWAP(list[i],list[min],temp);
   }
}
```

프로그램 1.4: 선택 정렬

```
void insert(element e, element a[], int i)
{/* insert e into the ordered list a[1:i] such that the
    resulting list a[1:i+1] is also ordered, the array a
    must have space allocated for at least i+2 elements */
   a[0] = e;
   while (e.key < a[i].key)
   {
      a[i+1] = a[i];
      i--;
   }
   a[i+1] = e;
}
```

프로그램 7.4: 정렬된 리스트로 삽입

```
void insertionSort(element a[ ], int n)
{ /* a[ 1:n] 을 비감소 키 순서대로 정렬 */
   int j;
   for (j = 2; j <= n; j++) {
      element temp = a [ j] ;
      insert(temp, a, j-1);
   }
}
```

프로그램 7.5: 삽입 정렬

```
void quickSort(element a[], int left, int right)
{/* sort a[left:right] into nondecreasing order
    on the key field; a[left].key is arbitrarily
    chosen as the pivot key;  it is assumed that
    a[left].key <= a[right+1].key */
  int pivot,i,j;
  element temp;
  if (left < right) {
     i = left; j = right + 1;
     pivot = a[left].key;
     do {/* search for keys from the left and right
            sublists, swapping out-of-order elements until
            the left and right boundaries cross or meet */
        do i++; while (a[i].key < pivot);
        do j--; while (a[j].key > pivot);
        if (i < j) SWAP(a[i],a[j],temp);
     } while (i < j);
     SWAP(a[left],a[j],temp);
     quickSort(a,left,j-1);
     quickSort(a,j+1,right);
  }
}
```

프로그램 7.6: 퀵 정렬