```c
int *input_array_dyn(int n)
{
	int i;
	int *a;
	a = (int *)malloc(n* sizeof(int));
	assert(a);   /* malloc() worked */
	printf("enter the array of length %d\n", n);
	for (i = 0; i<n; i++)
		scanf("%d", a + i);
	return a;
}

void array_print(int* a, int n)
{
	int i;
	for (i = 0; i<n; i++)
		printf("%d ", a[i]);
	printf("\n");
}

void swap(int *v, int *u)
{
	int temp;
	temp = *v;
	*v = *u;
	*u = temp;
}

void merge(int *a, int p, int q, int r)
{
	int i = p, j = q + 1, k = 0;
	int* temp = (int*)malloc((r - p + 1)* sizeof(int));

	while ((i <= q) && (j <= r))
		if (a[i]<a[j])
			temp[k++] = a[i++];
		else
			temp[k++] = a[j++];
	while (j <= r)
		temp[k++] = a[j++];
	while (i <= q)
		temp[k++] = a[i++];
		/* copy temp[] to a[]   */
	for (i = p, k = 0; i <= r; i++, k++)
		a[i] = temp[k];
	free(temp);
}

void merge_sort(int *a, int first, int last)
{
	int middle;
	if (first < last) {
		middle = (first + last) / 2;
		merge_sort(a, first, middle);
		merge_sort(a, middle + 1, last);
		merge(a, first, middle, last);
	}
}
```

```
int split(int *a, int left, int right)
{
	int i, last = left;
	if (left<right)
	{
		for (i = left + 1; i <= right; i++)
			if (a[i] <= a[left])
				swap(a + (++last), a + i);
		swap(a + left, a + last);
	}
	return last;
}

void quick_sort(int *x, int first, int last)
{
	int pos;
	if (first<last)
	{
		pos = split(x, first, last);
		quick_sort(x, first, pos - 1);
		quick_sort(x, pos + 1, last);
	}
}
```