```c
/* *** Codifique aqui a funcao bubbleSort versão 1 *** */
void bubbleSort (int *vetor, int N) {
    int i, aux, troca = 1;
    while (troca==1) {
        troca = 0;
        for (i = 0; i < N-1; i++) {
            if (vetor[i] > vetor[i+1]) {
                aux = vetor[i];
                vetor[i] = vetor[i+1];
                vetor[i+1] = aux;
                troca = 1;
            }
        }
    }
}
```

```c
/* *** Codifique aqui a funcao bubbleSort versão 2 *** */
void bubbleSort2 (int *vetor, int N) {
    int i, j, aux, troca = 1;
    for (i = 0; (i < N-1) && troca; i++) {
        troca = 0;
        for (j = 0; j < N-i-1; j++) {
            if (vetor[j] > vetor[j+1]) {
                aux = vetor[j];
                vetor[j] = vetor[j+1];
                vetor[j+1] = aux;
                troca = 1;
            }
        }
    }
}
```

```c
/* *** Codifique aqui a funcao selectSort *** */
void selectSort (int *vetor, int N) {
    int menor, aux, i, j;
    for (i = 0; i < N-1; i++) {
        menor = i;

        for (j = i+1; j < N; j++)
            if (vetor[j] < vetor[menor])
                menor = j;

        if (menor != i) {
            aux = vetor[i];
            vetor[i] = vetor[menor];
            vetor[menor] = aux;
        }
    }
}
```

```c
/* *** Codifique aqui a funcao insertSort *** */
void insertSort (int *vetor, int N) {
    int aux, i, j;
    for (i = 1; i < N; i++) {
        aux = vetor[i];

        for (j = i-1; (j >= 0) && (aux < vetor[j]); j--)
            vetor[j+1] = vetor[j];

        vetor[j+1] = aux;
    }
}
```

```c
int main (void) {

    int *base;
    int *ordenado;

    int n, i;

    /* gerando dados para teste */
    n = 10;
    base = (int*) malloc (n * sizeof(int));
    ordenado = (int*) malloc (n * sizeof(int));

    for (i = 0; i < n; i++)
        base[i] = rand() % 80;

    /* testando Bubble sort */
    printf("\nbubble sort version 1:\n");
    copyVector(base, ordenado, n);
    printVector(ordenado, n);
    bubbleSort(ordenado, n);
    printVector(ordenado, n);

    printf("\nbubble sort version 2:\n");
    copyVector(base, ordenado, n);
    printVector(ordenado, n);
    bubbleSort2(ordenado, n);
    printVector(ordenado, n);

    /* testando Select sort */
    printf("\nselect sort:\n");
    copyVector(base, ordenado, n);
    printVector(ordenado, n);
    selectSort(ordenado, n);
    printVector(ordenado, n);

    /* testando Insert sort */
    printf("\ninsert sort:\n");
    copyVector(base, ordenado, n);
    printVector(ordenado, n);
    insertSort(ordenado, n);
    printVector(ordenado, n);

    printf("\n");
    free(base);
    free(ordenado);
    return 0;
}
```