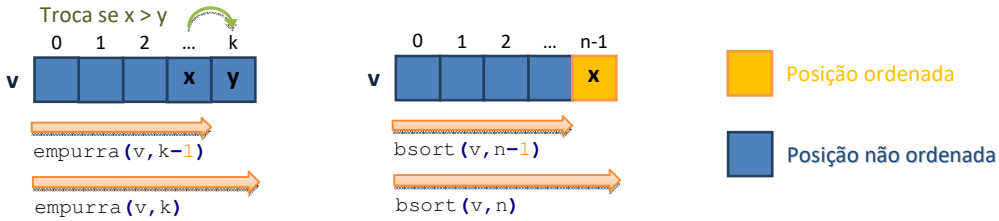
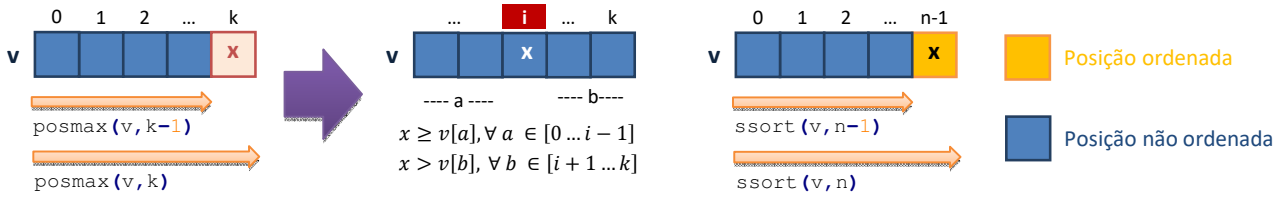
Faculdade de Tecnologia de São Paulo
Departamento de Tecnologia da Informação
Prof. Dr. Silvio do Lago Pereira
Aux. Doc. Lucio Nunes de Lira

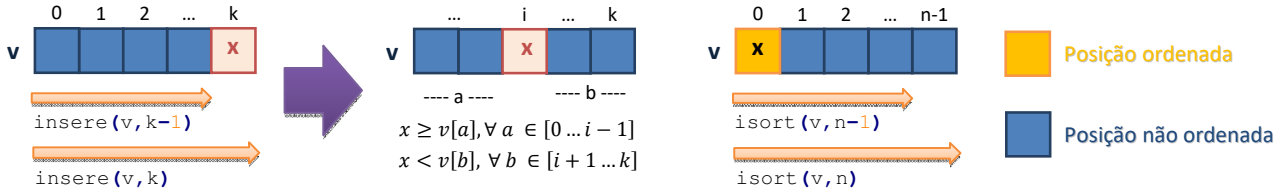
1. Crie a função recursiva `empurra(v,k)`, que empurra o maior item existente no vetor de inteiros `v[0..k]` para a posição `k` desse vetor. Em seguida, usando essa função e a ideia do *Bubble Sort*, crie a função recursiva `bsort(v,n)`, que ordena um vetor de inteiros `v[0..n-1]` em ordem crescente (como ilustrado a seguir).



2. Crie a função recursiva `posmax(v,k)`, que devolve a posição do maior item existente no vetor de inteiros `v[0..k]`. Em seguida, usando essa função e a ideia do *Selection Sort*, crie a função recursiva `ssort(v,n)`, que ordena um vetor de inteiros `v[0..n-1]` em ordem crescente (como ilustrado a seguir).



3. Crie a função recursiva `insere(x,v,k)`, que insere um item `x` num vetor de inteiros `v[0..k]`. Em seguida, usando essa função e a ideia do *Insertion Sort*, crie a função recursiva `isort(v,n)`, que ordena um vetor de inteiros `v[0..n-1]` em ordem crescente (como ilustrado a seguir).



4. Crie a função recursiva `lsearch(x,v,n)`, que faz uma busca linear (*linear search*) no vetor de inteiros *arbitrário* `v[0..n-1]` e devolve `1` se o item `x` estiver armazenado em `v` (ou `0`, caso contrário).

```c
#include <stdio.h>
#define MAX 10
int lsearch(int x,int v[],int n);
int main(void) {
   int v[MAX] = {88,31,96,52,45,19,68,70,29,90};
   printf("%d\n",lsearch(25, v, MAX));
   printf("%d\n",lsearch(70, v, MAX));
   return 0;
}
```

5. Crie a função recursiva `bsearch(x,v,i,f)`, que faz uma busca binária (*binary search*) no vetor de inteiros *ordenado* `v[i..f]` e devolve `1` se o item `x` estiver armazenado em `v` (ou `0`, caso contrário).

```c
#include <stdio.h>
#define MAX 10
int bsearch(int x,int v[],inti,int f);
int main(void) {
   int v[MAX] = {19,29,31,45,52,68,70,88,90,96};
   printf("%d\n",bsearch(25, v,0, MAX-1));
   printf("%d\n",bsearch(70, v,0, MAX-1));
   return 0;
}
```