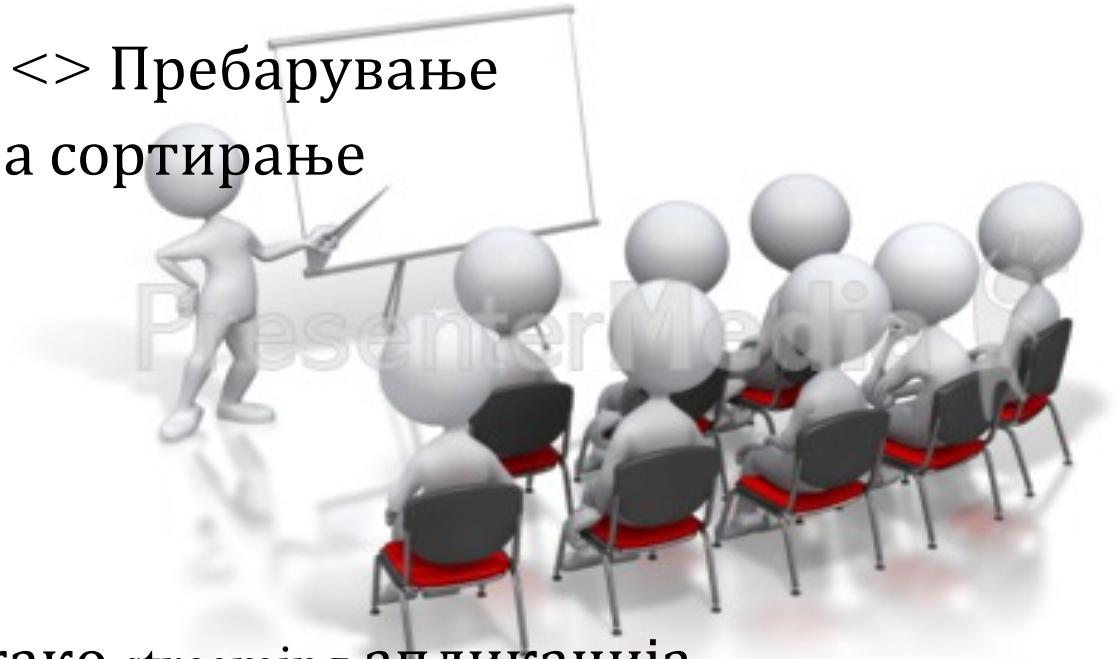
1. Алгоритми за сортирање
Напредно програмирање

Д-р Рамона Маркоска, вонр.проф.

Ставки за проучување.. во ова поглавје!



- Сортирање ? <> Пребарување
- Алгоритми за сортирање
- Bubble Sort
- Selection Sort
- Insertion Sort
- Quick Sort
- Selection Sort
- Сортирање како streaming апликација
- Споредба на разните алгоритми за сортирање
- Енумерација, Унии,
- Користење на printf, scanf, корелација со референцирање.

Алгоритми за сортирање

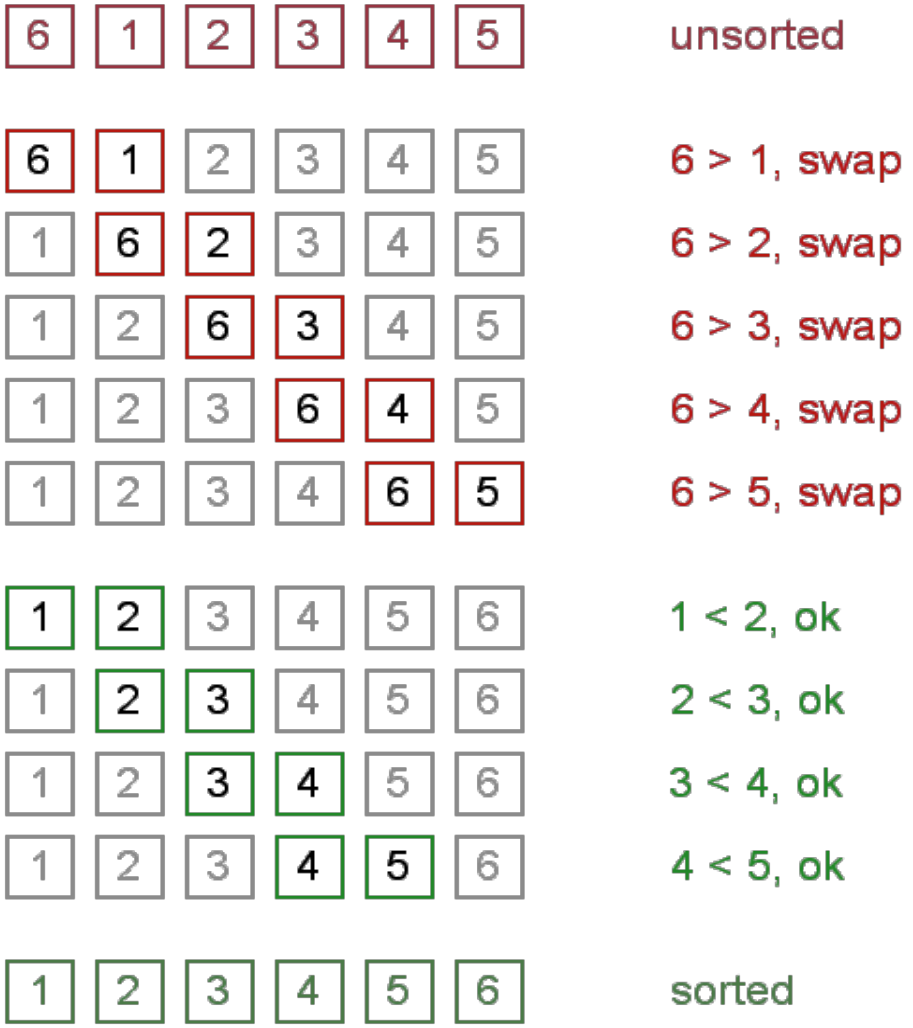
- Воведен дел- потсетување за низи
- Споредба на постапките на сортирање и пребарување, сличности и разлики

- Алгоритми за сортирање
  - Физикално значење во програмските решенија
  - Зошто има потреба од повеќе алгоритми
  - Класификација на алгоритмите
    - Базирани на компарација
    - Без компарација

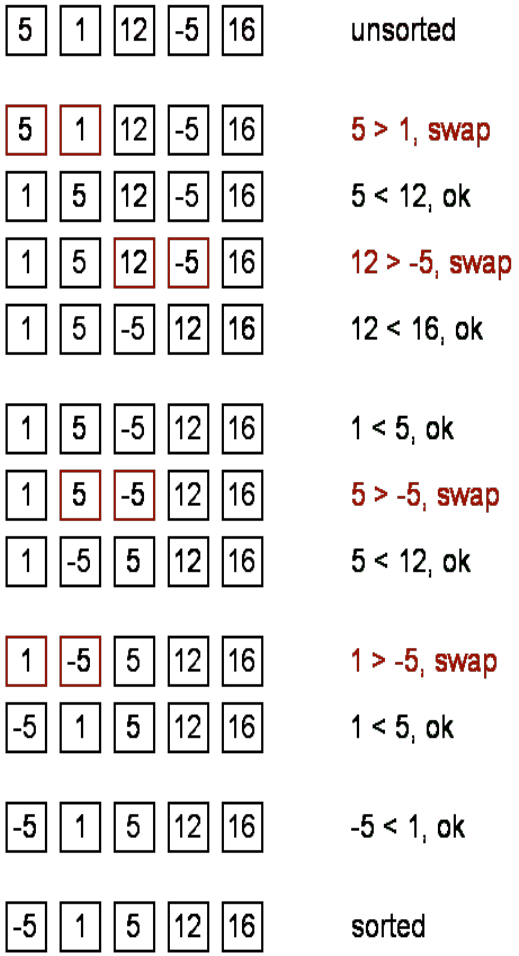
Алгоритми, манипулација со низи- сортирање Bubble Sort

Споредба меѓу сортирање и пребарување,

Bubble Sort- појаснување



Алгоритми, манипулација со низи- сортирање- Bubble Sort



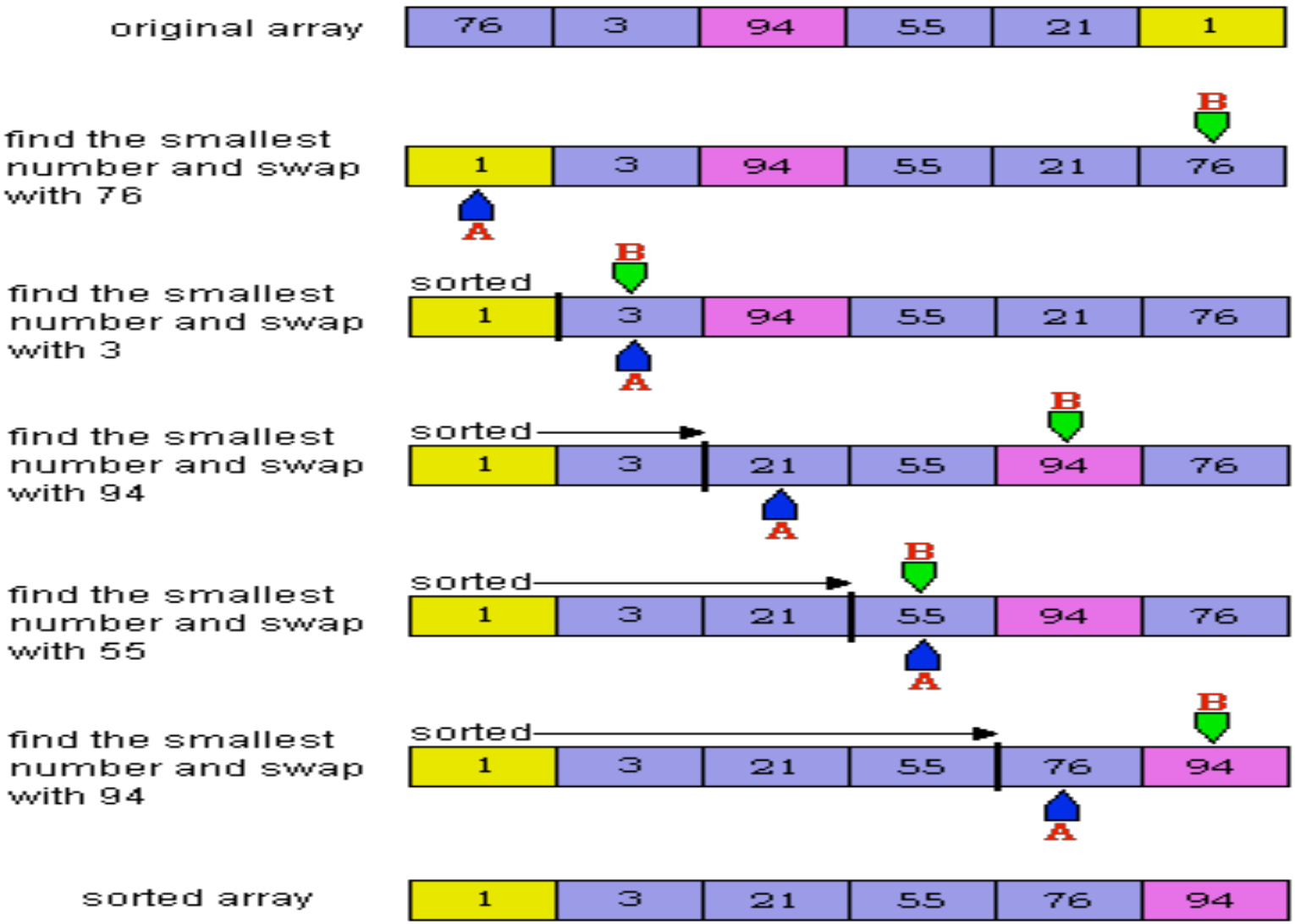
C++

```
void bubbleSort(int arr[], int n) {
      bool swapped = true;
      int j = 0;
      int tmp;
      while (swapped) {
            swapped = false;
            j++;
            for (int i = 0; i < n - j; i++) {
                  if (arr[i] > arr[i + 1]) {
                        tmp = arr[i];
                        arr[i] = arr[i + 1];
                        arr[i + 1] = tmp;
                        swapped = true;
                  }
            }
      }
}
```

Алготитми за сортирање- Selection Sort

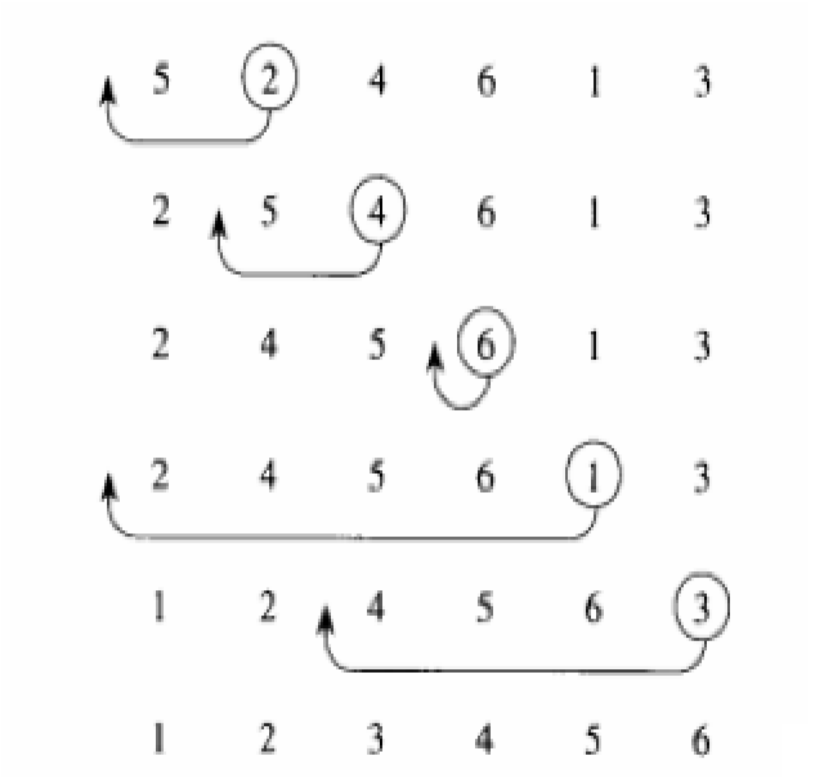
```
int A[SIZE] = {54,15,32,78,23,90,48,86,23,65};
int i, j, min, temp;

for (i = 0; i < SIZE - 1; i++){
    min = i;
    // find the minimum number in the array
    for (j = i+1; j < SIZE; j++){
        if (A[j] < A[min]){
            min = j;
        }
    }
    // swap the two numbers
    temp = A[i];
    A[i] = A[min];
    A[min] = temp;
}
```

Алготитми за сортирање- Selection Sort



Алгоритми за сортирање- Insertion Sort

```
for(i=1; i<n; i++){
        j=i;
        t=a[j];
        while (j>0 && a[j-1] > t){
                a[j]=a[j-1];
                j--;
        }
        a[j]=t;
}
```



Сл. 2 Имплементација на Insertion sort во C++ и пример за операциите кои ги прави овај алгоритам на низата (5, 2, 4, 6, 1, 3)

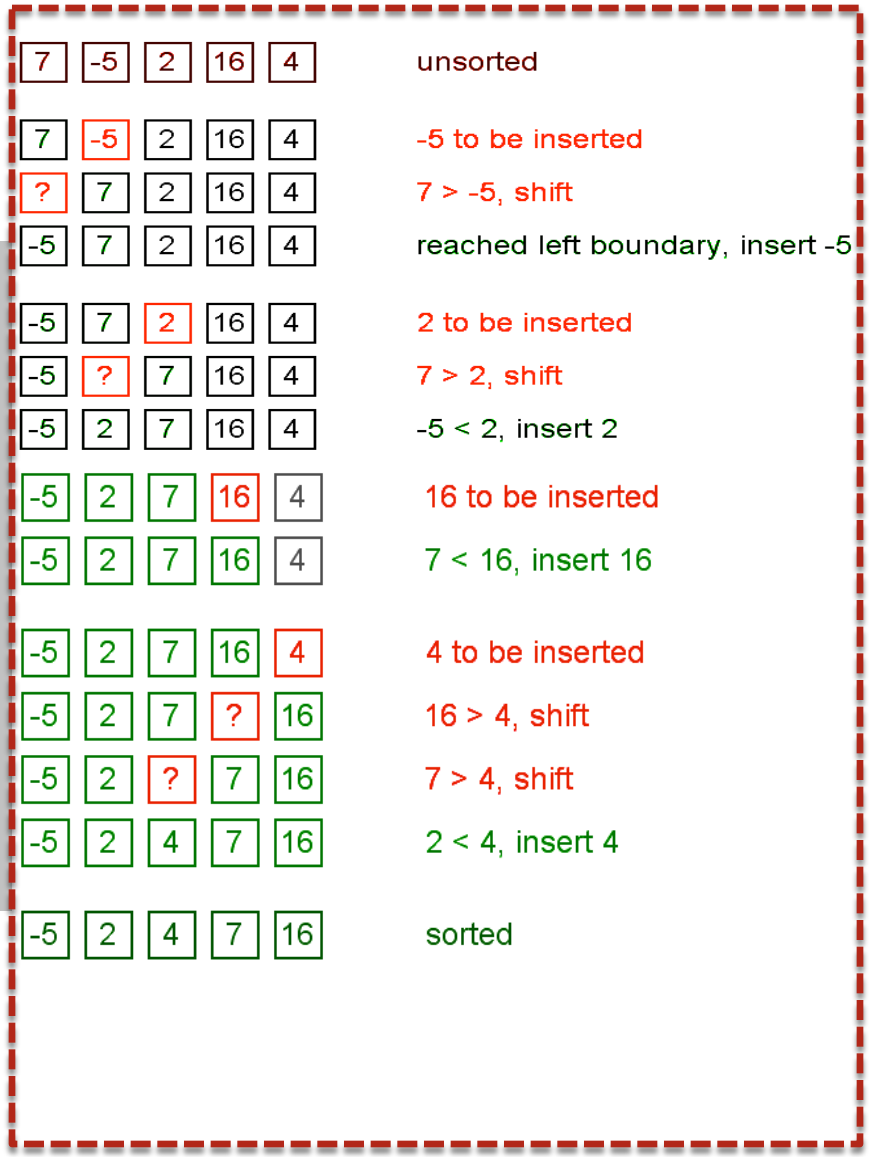
Алгоритми за сортирање- Insertion Sort

На сл.2 е даден пример којшто ги кажува операциите кои ги прави insertion sort на низата (5, 2, 4, 6, 1, 3). Со кругче ни е означена позицијата на индексот j, и елементот којшто треба да го поставиме на точната позиција. Лево од него се наоѓа веќе сортираниот дел од низата додека десно се останатите елементи кои треба да се сортираат. Пример во вториот ред, го имаме бројот 4 којшто треба да го поставиме на точна позиција. Го споредуваме со 5 и 2 (броевите лево од него, кои се веќе сортирани) и ја откриваме неговата точната позиција која е помегу 2 и 5. Со стрелка ни е означена позицијата каде што треба да биде поставен секој елемент. На крајот како резултат ја добиваме целосно сортираната низа.

Алгоритми за сортирање- Insertion Sort

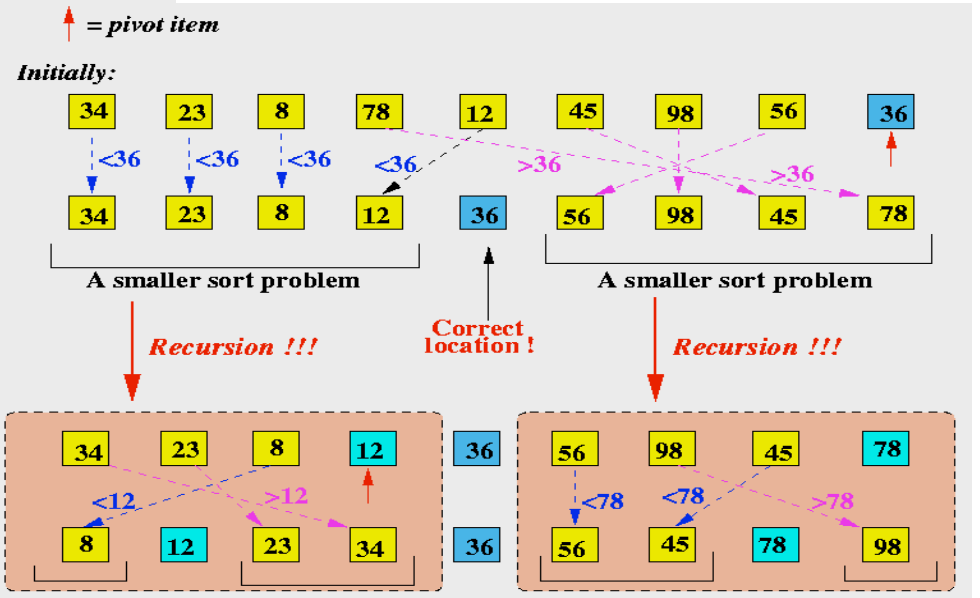
C++ implementation

```
void insertionSort(int arr[], int length) {
      int i, j, tmp;
      for (i = 1; i < length; i++) {
            j = i;
            while (j > 0 && arr[j - 1] > arr[j]) {
                  tmp = arr[j];
                  arr[j] = arr[j - 1];
                  arr[j - 1] = tmp;
                  j--;
            }
      }
}
```



Алгоритми за сортирање-Quick Sort

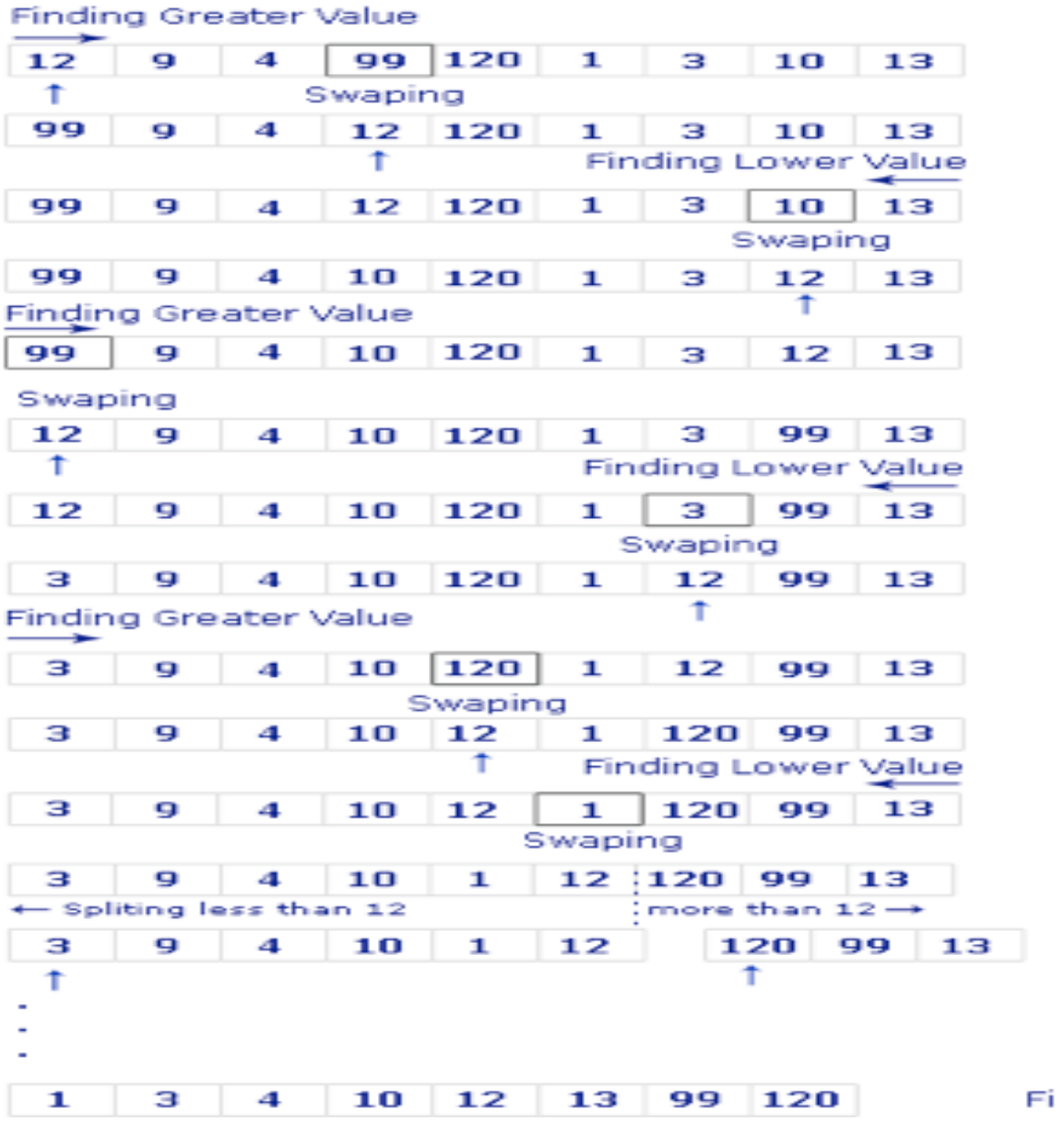
```
int partition(int a[], int left, int right, int pivotIndex)
{
    int pivot = a[pivotIndex];
    do
    {
        while (a[left] < pivot) left++;
        while (a[right] > pivot) right--;
        if (left < right && a[left] != a[right])
        {
            swap(a[left], a[right]);
        }
        else
        {
            return right;
        }
    }
    while (left < right);
    return right;
}
void quicksort(int a[], int left, int right)
{
    if (left < right)
    {
        int pivot = (left + right) / 2; // middle
        int pivotNew = partition(a, left, right, pivot);
        quicksort(a, left, pivotNew - 1);
        quicksort(a, pivotNew + 1, right);
    }
}
```



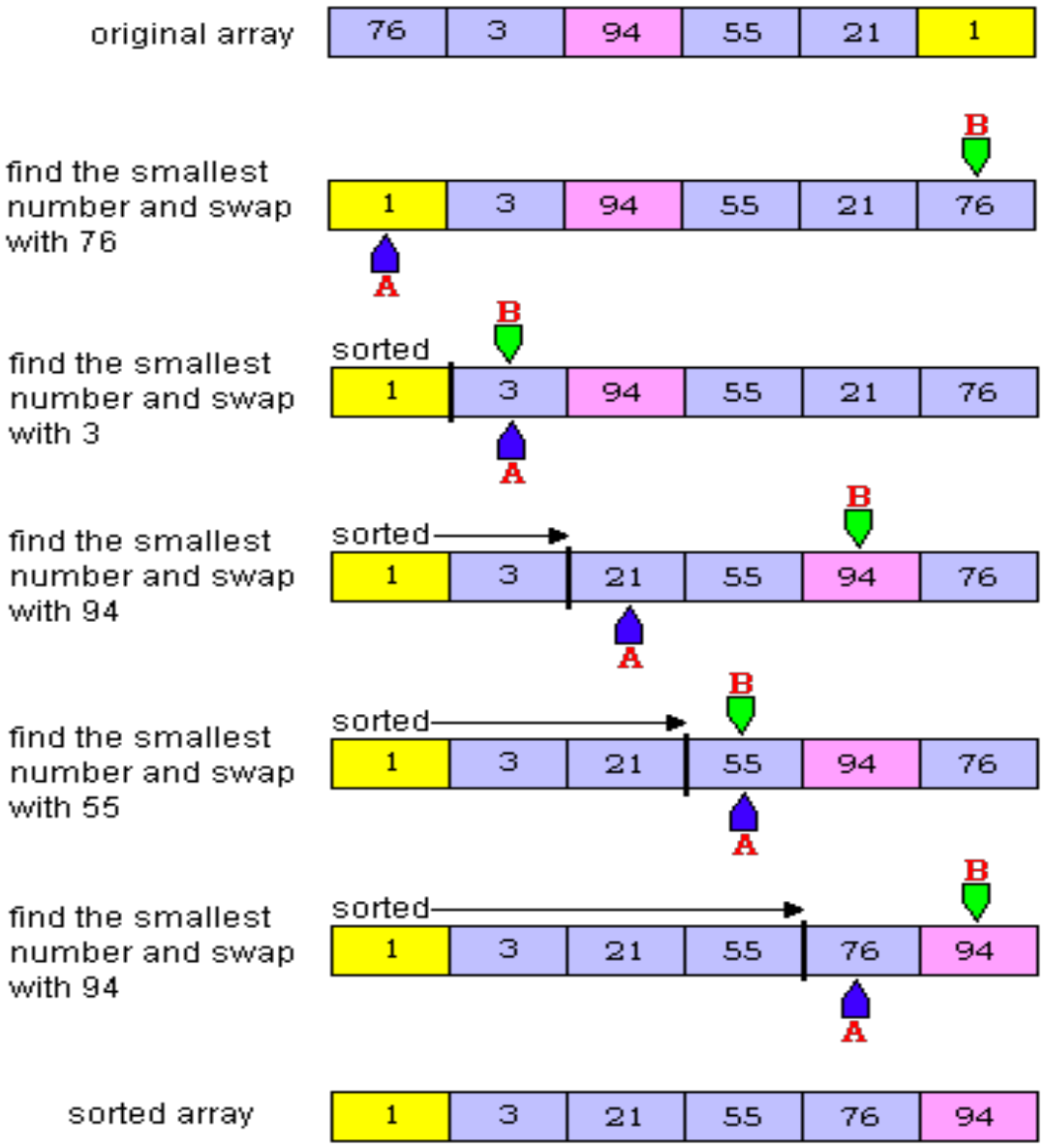
Алгоритми за сортирање- Quick Sort

Објаснување на алгоритам

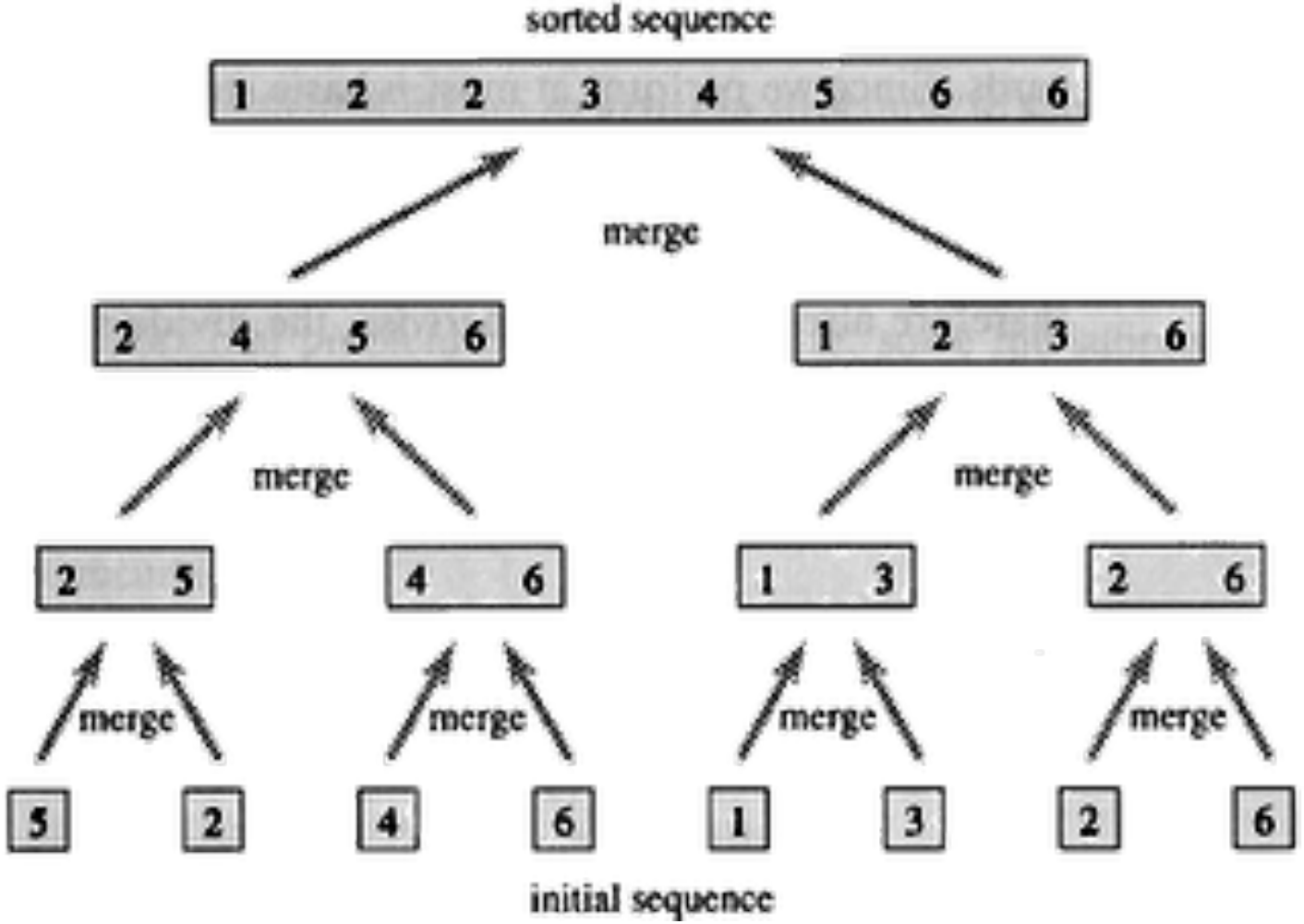
- Поголема од десно
- Замена
- Помала од лево
- Замена



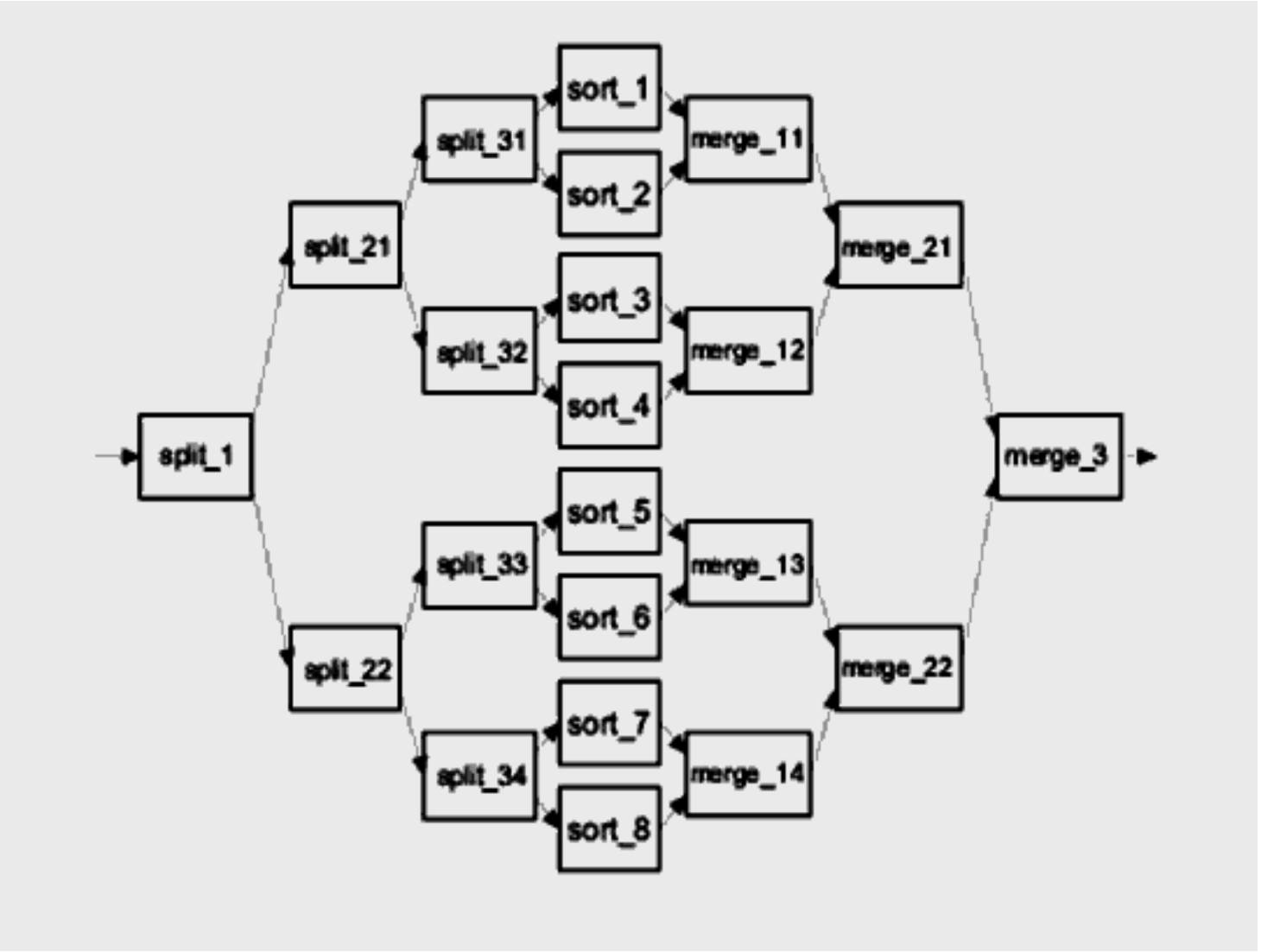
Алгоритми за сортирање- Selection Sort



Алгоритми за сортирање- Merge Sort



Сортирање како стриминг апликација



Споредба на разни алгоритми за сортирање

		Time Complexity					
		Best	Worst	Avg.	Space	Stable	Comments
Comparison Sort	Bubble Sort	O(n^2)	O(n^2)	O(n^2)	O(1)	Yes	*For each pair of indices, swap the elements if they are out of order*
	Modified Bubble Sort	O(n)	O(n^2)	O(n^2)	O(1)	Yes	*At each Pass check if the Array is already sorted. Best Case-Array Already sorted*
	Selection Sort	O(n^2)	O(n^2)	O(n^2)	O(1)	Yes	*Swap happens only when once in a Single pass*
	Insertion Sort	O(n)	O(n^2)	O(n^2)	O(1)	Yes	*Very small constant factor even if the complexity is O(n^2).* ***Best Case:*** *Array already sorted* ***Worst Case:*** *sorted in reverse order*
	Quick Sort	O(n.lg(n))	O(n^2)	O(n.lg(n))	O(1)	Yes	*Best Case: when pivot divide in 2 equal halves Worst Case: Array already sorted - 1/n-1 partition*
	Randomized Quick Sort	O(n.lg(n))	O(n.lg(n))	O(n.lg(n))	O(1)	Yes	*Pivot chosen randomly*
	Merge Sort	O(n.lg(n))	O(n.lg(n))	O(n.lg(n))	O(n)	Yes	*Best to sort linked-list (constant extra space). Best for very large number of elements which cannot fit in memory (External sorting)*
	Heap Sort	O(n.lg(n))	O(n.lg(n))	O(n.lg(n))	O(1)	No	
Non-Comparison Sort	Counting Sort	O(n+k)	O(n+k)	O(n+k)	O(n+2^k)	Yes	*k = Range of Numbers in the list*
	Radix Sort	O(n.k/s)	O(2^s.n.k/s)	O(n.k/s)	O(n)	No	
	Bucket Sort	O(n.k)	O(n^2.k)	O(n.k)	O(n.k)	Yes	

MCN Professionals: *Training & Placement division of* Mindcracker *Network*

C- <cstdio> printf,scanf -примена

```c
/* printf example */
#include <stdio.h>

int main()
{
   printf ("Characters: %c %c \n", 'a', 65);
   printf ("Decimals: %d %ld\n", 1977, 650000L);
   printf ("Preceding with blanks: %10d \n", 1977);
   printf ("Preceding with zeros: %010d \n", 1977);
   printf ("Some different radixes: %d %x %o %#x %#o \n", 100, 100, 100,
   printf ("floats: %4.2f %+.0e %E \n", 3.1416, 3.1416, 3.1416);
   printf ("Width trick: %*d \n", 5, 10);
   printf ("%s \n", "A string");
   return 0;
}
```

```
Characters: a A
Decimals: 1977 650000
Preceding with blanks:       1977
Preceding with zeros: 0000001977
Some different radixes: 100 64 144 0x64 0144
floats: 3.14 +3e+000 3.141600E+000
Width trick:    10
A string
```

Scanf- примена

```c
/* scanf example */
#include <stdio.h>

int main ()
{
  char str [80];
  int i;

  printf ("Enter your family name: ");
  scanf ("%s",str);
  printf ("Enter your age: ");
  scanf ("%d",&i);
  printf ("Mr. %s , %d years old.\n",str,i);
  printf ("Enter a hexadecimal number: ");
  scanf ("%x",&i);
  printf ("You have entered %#x (%d).\n",i,i);

  return 0;
}
```

This example demonstrates some of the types that can be read with scanf:

```
Enter your family name: Soulie
Enter your age: 29
Mr. Soulie , 29 years old.
Enter a hexadecimal number: ff
You have entered 0xff (255).
```

Кратенки- значење

Табеларен преглед

\n	newline
\r	carriage return
\t	tab
\v	vertical tab
\b	backspace
\f	form feed (page feed)
\a	alert (beep)
\'	single quote (')
\"	double quote (")
\?	question mark (?)
\\	backslash (\)

Користење

- Најчесто се користат во комбинација со printf , scanf

Енумерација

- Специјално дефиниран сложен тип на податоци кој има свој:
  - сет на вредности
  - операции дефинирани врз тие вредности

Енумерација 2

-Дефиниција: Овозможува опишување на опсег на вредности и придружување на подредени константи

```
enum colors_t {black, blue, green, cyan, red, purple, yellow, white};
```

- Пример за разни операции

```
1 colors_t mycolor;
2
3 mycolor = blue;
4 if (mycolor == green) mycolor = red;
```

```
1 enum months_t { january=1, february, march, april,
2                 may, june, july, august,
3                 september, october, november, december} y2k;
```

Унии

Дефиниција, поента

- Ист дел меморија користен од разни типови на податоци
- Големината одговара на најголемиот елемент(променлива) од декларацијата
- Рационално користење на меморија
- Корелација со типовите на покажувачи

Пример

```
1 union mytypes_t {
2   char c;
3   int i;
4   float f;
5   } mytypes;
```

defines three elements:

```
1 mytypes.c
2 mytypes.i
3 mytypes.f
```

Материјали за учење и практични примери

Дел 2 од Збирка- комбинирано користење на printf, scanf, Алгоритми за сортирање

Клипови линкувани на сајт