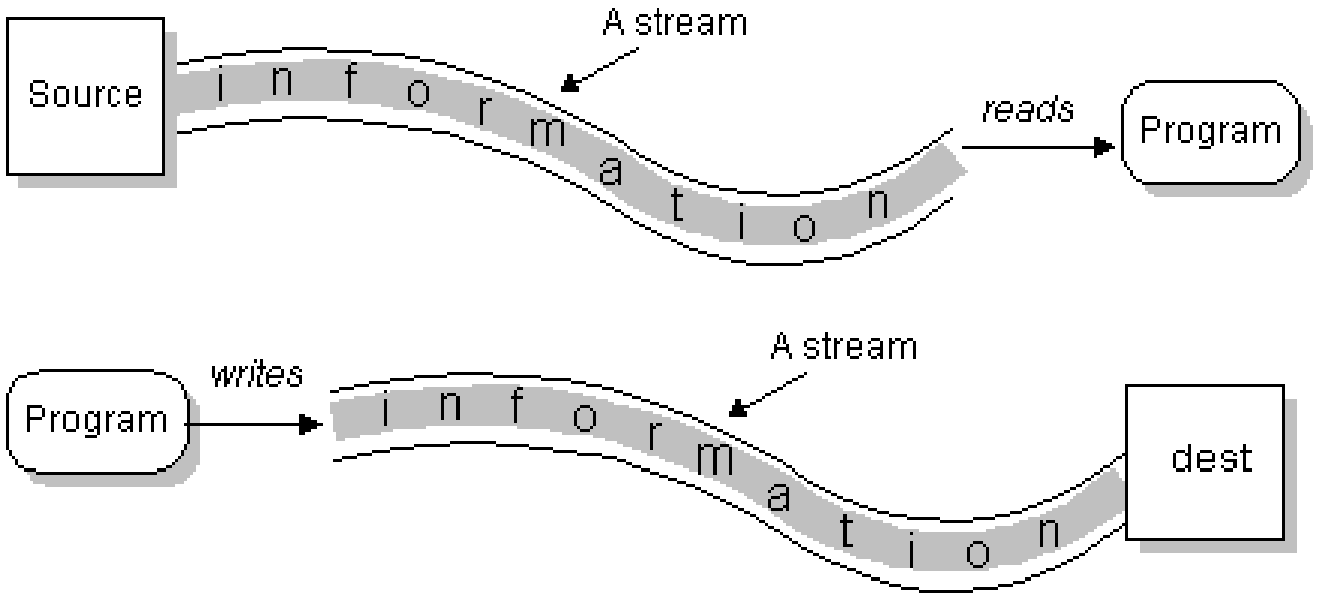
Файлы и деревья

Семестр 1
Семинар 11

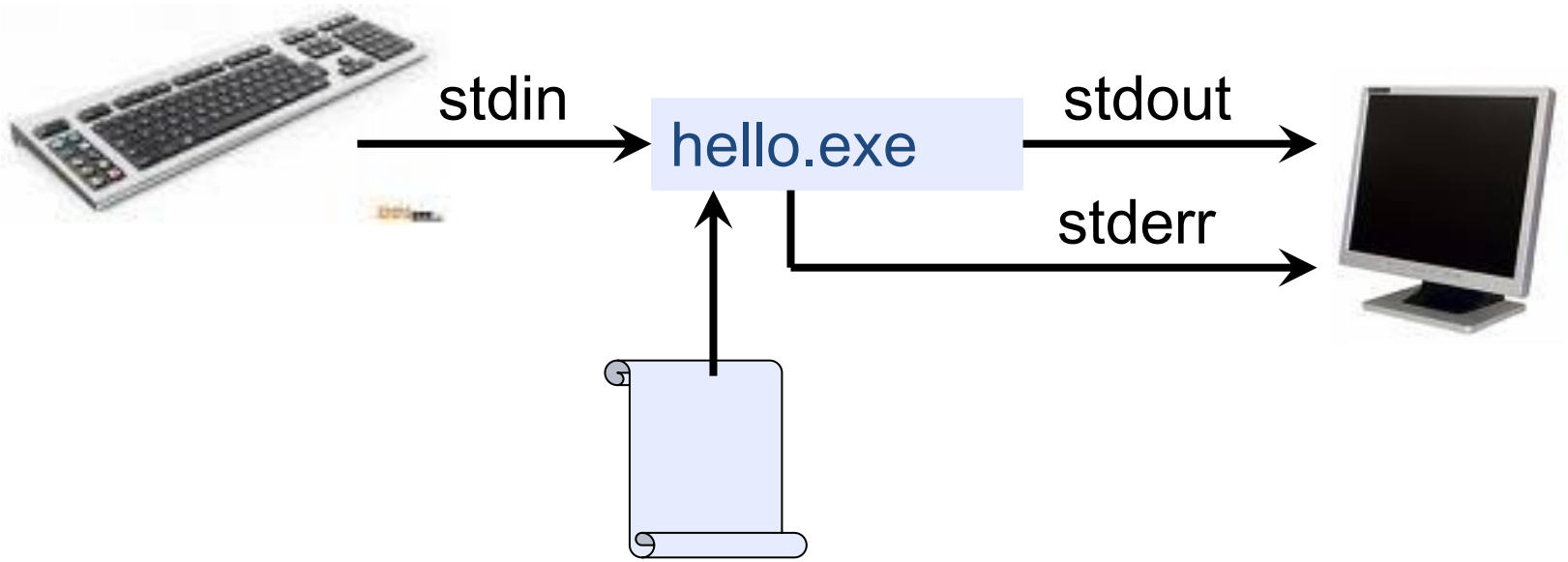
Stream (поток)

- stream - абстракция, которая или производит или потребляет информацию
- очередь



Потоки данных (stream)

- stdin (getchar, scanf)
- stdout (printf)
- stderr



Работа с потоком

```
FILE * fopen (char * file, char * mode);
int    fclose (FILE * stream);
```

file - путь к файлу

- абсолютный
  "/home/gr793/s69308/hello.c"
- относительный
  "../work/b.txt" "1.c"
- грабли Windows
  "c:\stud\natasha\hello.c" -> экранирующая последовательность

"c:\\stud\\natasha\\hello.c" -> вот так сработает

Режимы открытия файла

r	read	чтение
w	write	запись (обрезать длину до 0)
a	append	добавление (писать в конец)
r+	rw	дополнительные режимы
w+	wr	
a+	ra	
rb	**binary** *в UNIX* *ignore*	\r\n -> \n
wb		\n -> \r\n
ab		таких преобразований нет
rb+		комбинации и далее

Пример

Копирование посимвольное файла

```
#include <stdio.h>
int main() {
    int c;
    FILE * in  = fopen ("in.txt", "r");
    FILE * out = fopen ("out.txt", "w");
    while ( (c=fgetc(in)) != EOF )
       fputc(c,out);
    fclose (in);
    fclose (out);
  return 0;
}
```

Пример

Копирование посимвольное файла

```
#include<stdio.h>
#include<stdlib.h>
int main() {
    int c;
    FILE * in  = fopen ("in.txt", "r");
    FILE * out = fopen ("out.txt", "w");
    if (!in || !out) {
       printf("no file exist\n");
       return 0;
    }
    while ( (c = fgetc(in)) != EOF )
       fputc(c, out);
    fclose (in);
    fclose (out);
    printf("job's done\n");
    return 0;
}
```

Функции вывода

```
int   printf (char * format, …);
int  fprintf (FILE * fp, char * format, …);
        fprintf(out, "max is %d\n", max_arr);
int  sprintf (char * str, char * format, …);
        char string[1000]="";
        sprintf(string,"kdgkjhbg %d kdsrjg", c);
char * fputs (char * s, FILE * fp);
        fputs(string, out);
char * puts (char * s);
```

Функции ввода

```
int   scanf (char * format, …);
int  fscanf (FILE * fp, char * format, …);
        fscanf(in,"%d",&c);
int  sscanf (char * str, char * format, …);
        sscanf(string,"%d",&c);
char * fgets (char * s, int n, FILE * fp);
        fgets(string, 15, in);
```

Позиционирование в файле

- int fgetpos (FILE * fp, fpos_t * ptr);
- int fsetpos (FILE * fp, fpos_t * ptr);
- если в файл и читаете, и пишите, то надо завести два указателя - один для сохранения позиции чтения, другой - записи.

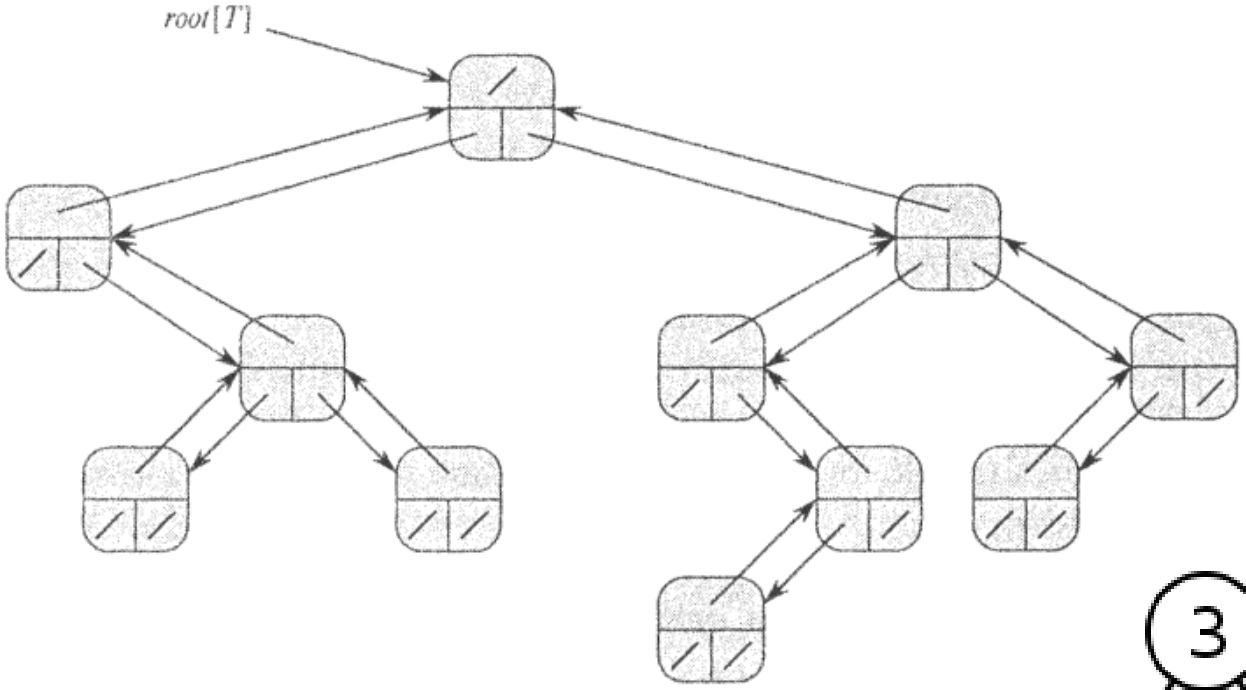
Позиционирование в файле

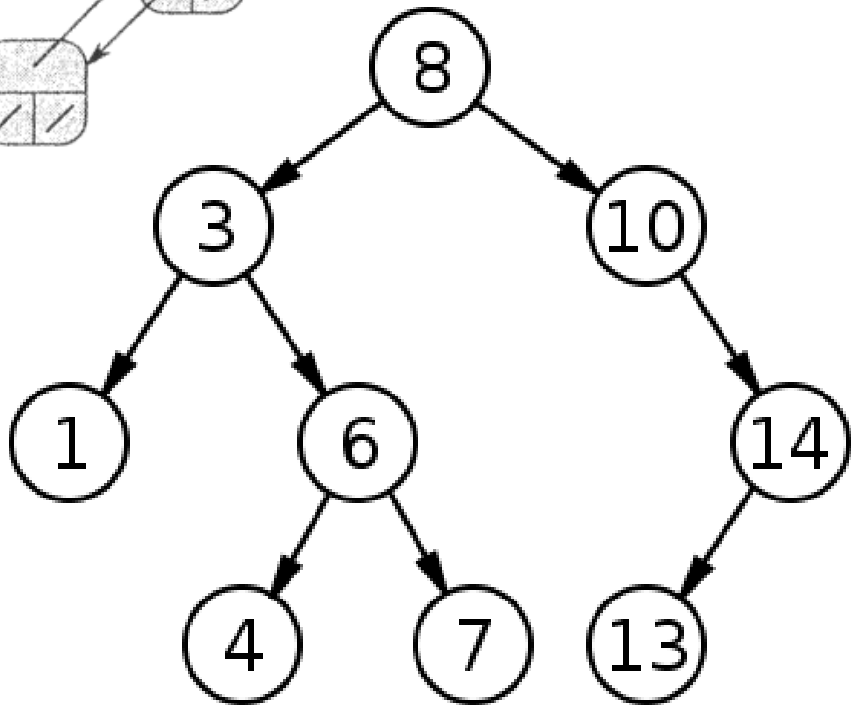
```
#include<stdio.h>
#include<stdlib.h>
int main()
{
    int c=55;
    char string[1000] =
              "awefaewrgaeg";
    FILE *in = fopen
              ("in.txt", "r");
    fpos_t in_ptr;
    if (!in)
    {
       printf("no file");
       return 0;
    }
    fgetpos(in,&in_ptr);
    printf("\n ptr in %d \n",
                    in_ptr);
    fgets(string, 10, in);
    puts(string);
    fgetpos(in, &in_ptr);
    printf("\n ptr in %d \n",
                    in_ptr);
    in_ptr = 5;
    fsetpos(in, &in_ptr);
    printf("\n ptr in %d \n",
                    in_ptr);
    fgets(string, 10, in);
    puts(string);
    fgetpos(in, &in_ptr);
    printf("\n ptr in %d \n",
                    in_ptr);
    putchar('\n');
    fclose (in);
    fclose (out);
    printf("job's done\n");
    return 0;
}
```

Задачи

0. Создать файл in.txt. Записать туда количество элементов массива и сам массив произвольных целых чисел.
1. Считать из файла in.txt количество элементов и массив чисел. Отсортировать их методом простого выбора и записать отсортированный массив в файл out.txt.
2. Забить все символы в файле out.txt символом ‘♫’.
3. Заполнить файл in.txt произвольным текстом из тысячи символов.
4. Считывая из файла in.txt по десять символов, обрабатывать их и выводить в файл out.txt. Символ ‘\n’ заменять на ‘n’.
   - сортировать
   - обращать порядок («1234567890» → «0987654321»)
   - заменять все цифры на соответствующие буквы («0» → «A»)

Бинарные деревья





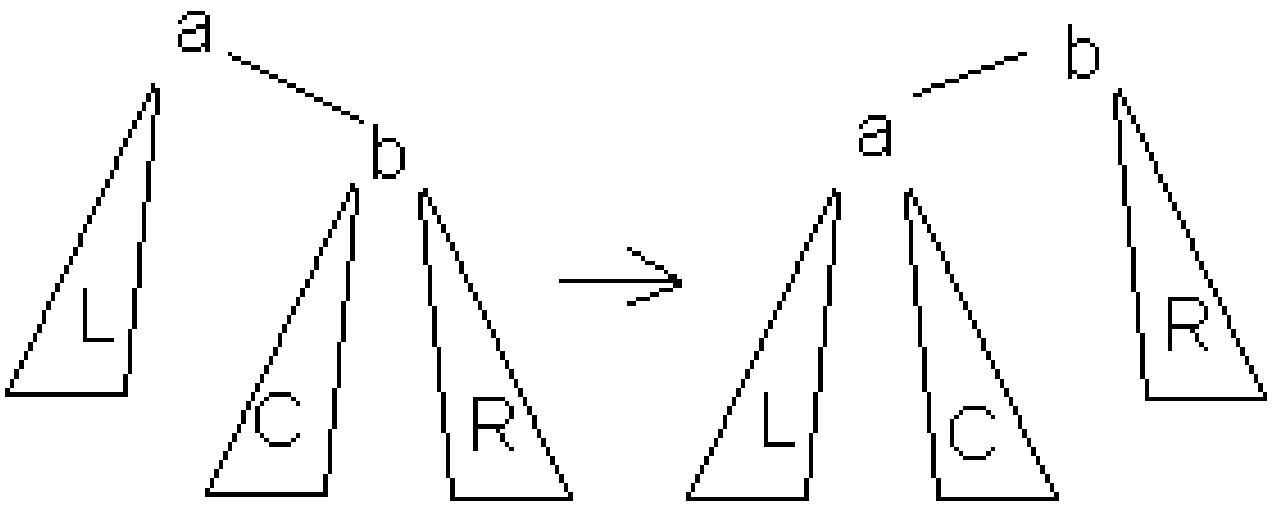
Дерево поиска

АВЛ-дерево

- сбалансированное по высоте двоичное дерево поиска: для каждой его вершины высота её двух поддеревьев различается не более чем на 1.
- АВЛ — аббревиатура, образованная первыми буквами создателей (советских учёных) Адельсон-Вельского Георгия Максимовича и Ландиса Евгения Михайловича.

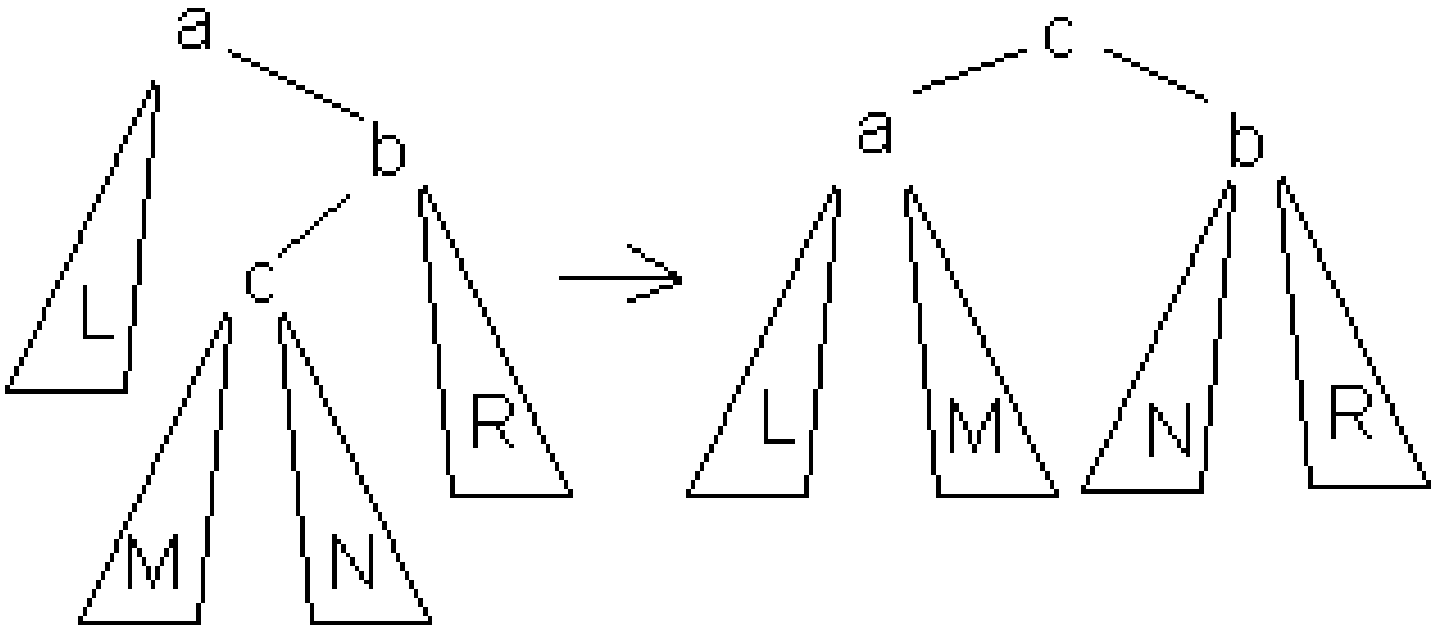
Малое левое вращение

- высота b-поддерева - высота L = 2
- высота C <= высота R



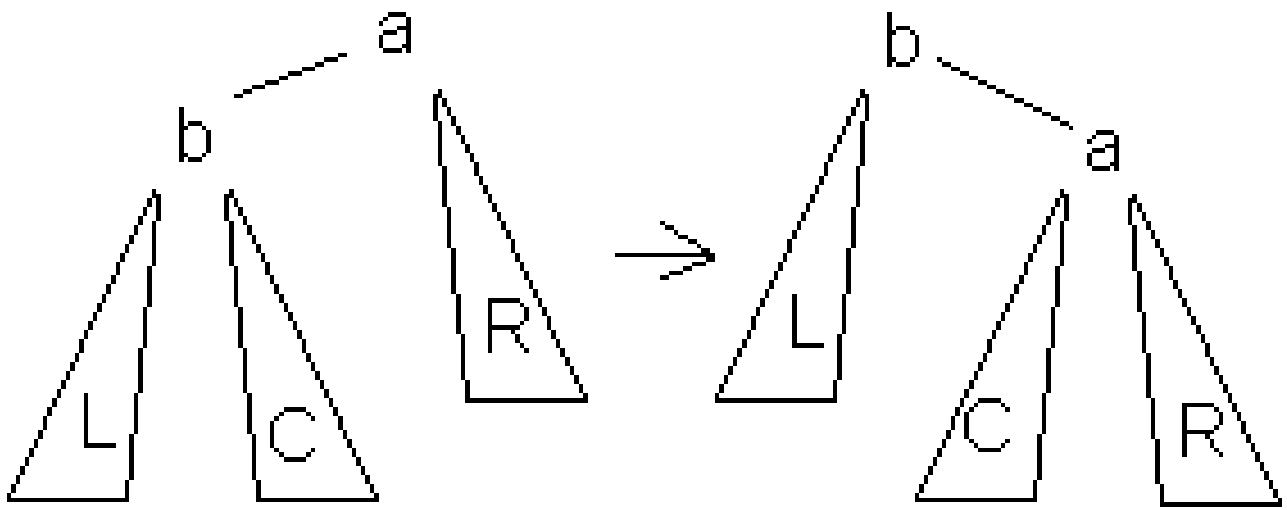
Большое левое вращение

- высота b-поддерева - высота L = 2
- высота C > высота R



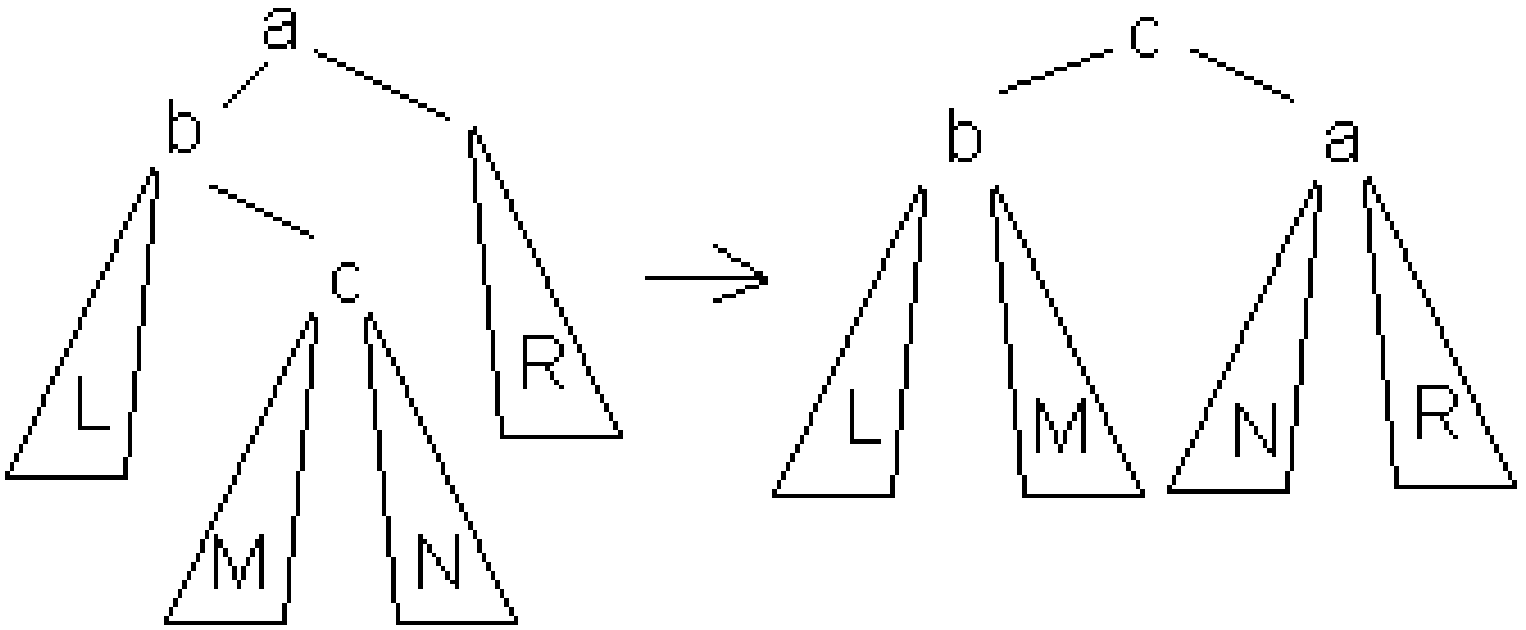
Малое правое вращение

- высота b-поддерева - высота R = 2
- высота C <= высота L



Большое правое вращение

- высота b-поддерева - высота R = 2
- высота C > высота L



Задачи

В каждом узле, помимо ссылок на потомков, содержится одно поле типа int и одно поле типа char*:

1) добавление элемента в дерево;

2) поиск элемента в дереве (по int);

3) удаление элемента из дерева (по int);

4) печать дерева;

5) обход дерева в глубину и ширину;

6) определение высоты дерева и поддерева;

7) одинарные повороты влево и вправо;

8) двойные повороты влево и вправо;

9) АВЛ-дерево (проверки высоты и вызов поворотов при добавлении и удалении элементов);

10*) визуализация.

Подключение файлов

```
//main.cpp
#include<stdio.h>
#include<d:\Student\add.cpp>
int main()
{
    int x=55, y=9;
    printf ("%d\n", sum(x,y));
    return 0;
}
```

```
//add.cpp
int sum (int x, int y)
{
    return x+y;
}
```