Лекция 4.
Файлы записей.
Структуры данных

Передача параметров функции main()

1. int main ( )
2. int main (int argc, char *argv[])

3. ./my_proga text1 text2

argc 3
argv[0] my_proga
argv[1] text1
argv[2] text2

Пример 1.

```
1.  #include<stdio.h>
2.  int main ( int argc, char *argv[])
3.  {
4.      int i=0;
5.      printf ("Число параметров равно%d\n",
    argc);
6.      printf ("Имя программы %s\n", argv[0]);
7.      for (i=1; i < argc; i++)
8.          printf (“аргумент %d равен %s\n", i, argv[i]);
9.      return 0;
10.   }
```

./my_proga myfile1.txt myfile2.txt myfile3.txt

Параметры функции main() в качестве имен файлов

```
f1=fopen(argv[1], “rb");
f2=fopen(argv[2], “wb");
```

Другие полезные функции

Функция **fseek( )** устанавливает файловый указатель в произвольную позицию внутри файла.

Прототип:

```
int fseek(FILE *fp, long count, int access);
```

fp - указатель на файл

Функция fseek( )

count - номер байта относительно начальной позиции, начиная с которого будет выполняться операция:

```
count = sizeof( type_of_object )*N;
```

Начальная позиция

int access

- способ задания начальной позиции:

0 - начальная позиция задана в начале файла;

1 - начальная позиция считается текущей;

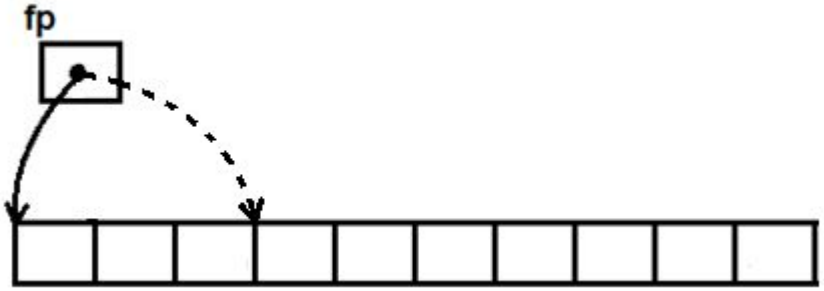
2 - начальная позиция задана в конце файла.

Прямой доступ.

```
struct data {
        int day;
        char month[10];
        int year;
            };
struct data    mydata, *dat;
dat = &mydata;
```

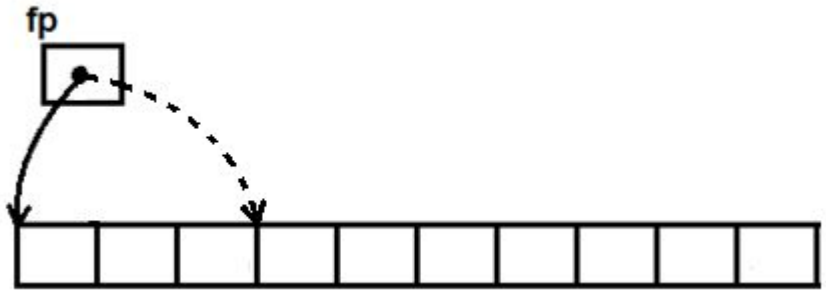
Способ задания начальной позиции

```
fseek(fp, sizeof(mydata)*3, 0);
fwrite(&MyData,sizeof(MyData), 1, fp);
```



Способ задания начальной позиции

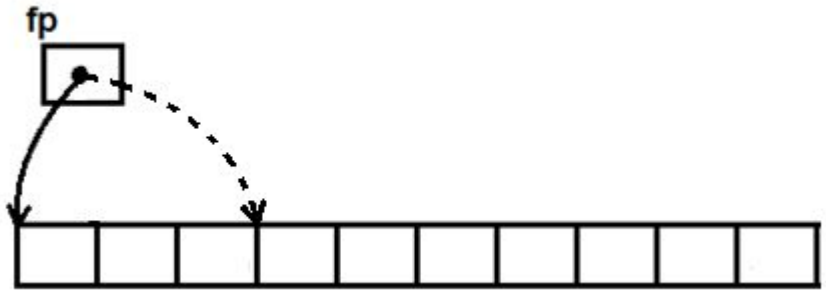
```
fread(&MyData, sizeof(mydata), 1, fp);
fseek(fp, sizeof(mydata)*2, 1);
fwrite(&mydata,sizeof(mydata), 1, fp);
```



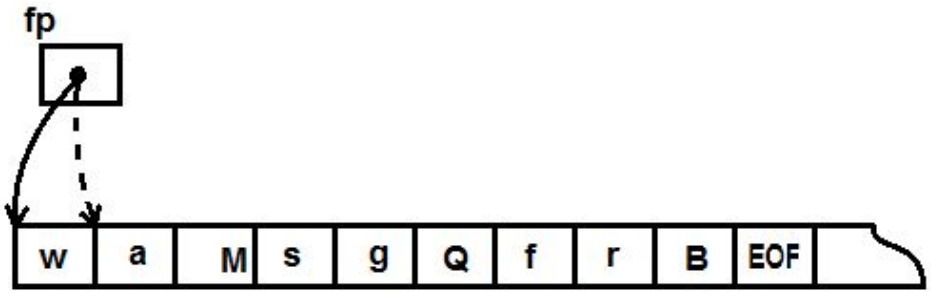
Способ задания начальной позиции

fseek(fp, sizeof(mydata)*(-7), **2**);

fwrite(&mydata,sizeof(mydata), 1, fp);



Последовательный и прямой доступ к записям файла




```
float a;
FILE *f;
f=fopen("numb","rb");
fscanf(f, "%f", &a);
fscanf(f, "%f", &a);
fscanf(f, "%f", &a);
printf("%f", a);
```

1.1 2.2 3.3 4.4 5.5 6.6

Пример 2. Прямой доступ. Обновление четвертой записи

```
1. FILE *fp;
2. fp=fopen("mystructs", "rb+");
3. fseek(fp, sizeof(mydata)*3, 0);
4. fread(&dat, sizeof(mydata), 1, fp);
5. scanf("%s", &mydata.month);
6. fseek(fp, sizeof(mydata)*3, 0);
7. fwrite(&mydata,sizeof(mydata), 1, fp);
```

Указатель текущей позиции в начало файла

- Функция rewind() :

```
void rewind (FILE *fp);
```

Следует использовать, если чтение файла должно быть выполнено несколько раз

Пример 3. Передача файлового указателя функции

```
1. void prosmotr(FILE *fp)
2. {    rewind(fp);
3. // использование fread(…)    }
4. int main()
5. {
6.     FILE *fp;
7.     fp=fopen("mystructs","rb+");
8.     prosmotr(fp);
9.     dobavlenie(fp); …
```

Структуры данных. Линейный список

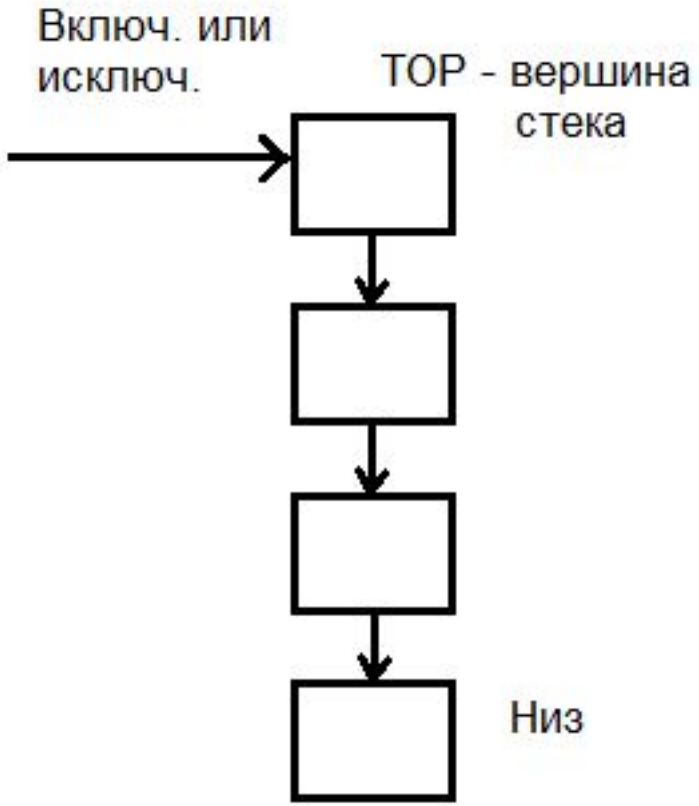
Линейный список - это множество, состоящее из переменного числа узлов

X[1], X[2], … , X[n], n≥0

1. Если n>0 X[1] – первый элемент
2. Если 1<k<n X[k-1], X[k+1]
3. X[n] - последний элемент

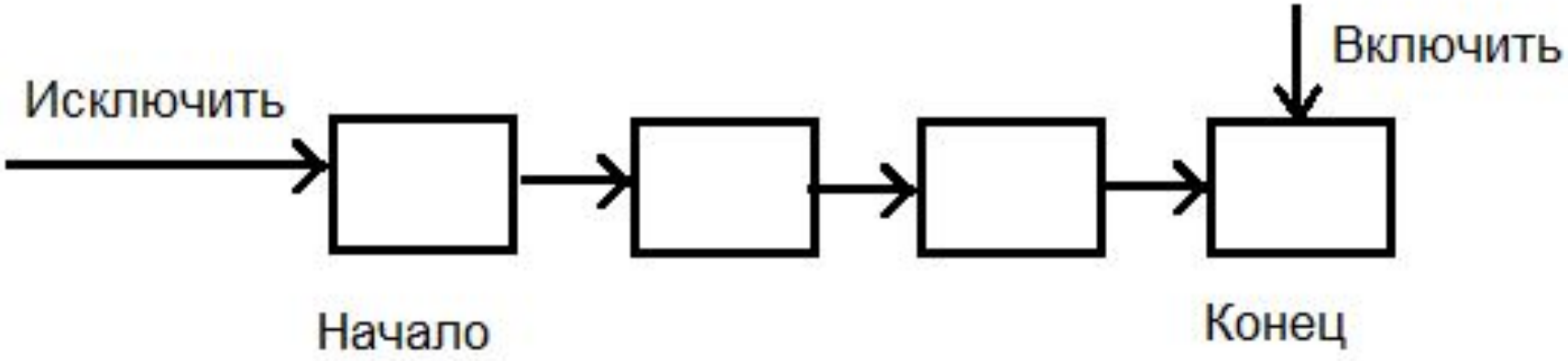
Некоторые специальные списки

Стек



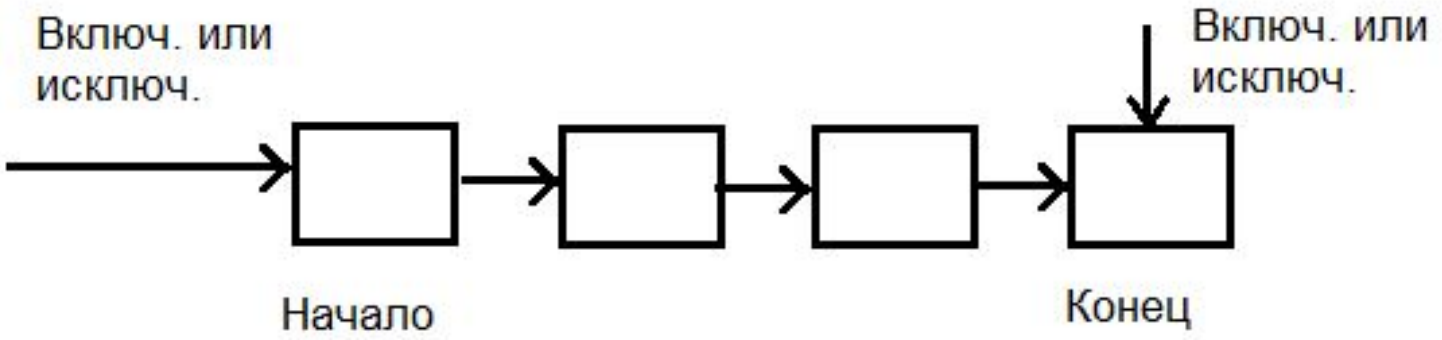
Некоторые специальные списки

Очередь



Некоторые специальные списки

Дек



Пример 4. Реализация стека в виде массива

```
1.  include<stdio.h>
2.   int Stack[100], TOP=0 ;
3.   void vcluch(int);
4.   viod iskluch(*int);
```

Реализация стека в виде массива. Функция включения элемента в стек

```
1.    void vcluch (int y)
2.     {
3.        if( TOP>=100 )
4.        {   printf(“Переполнение стека”);
5.            return;  }
6.     Stack[TOP] = y;
7.     TOP++;
8.    }
```

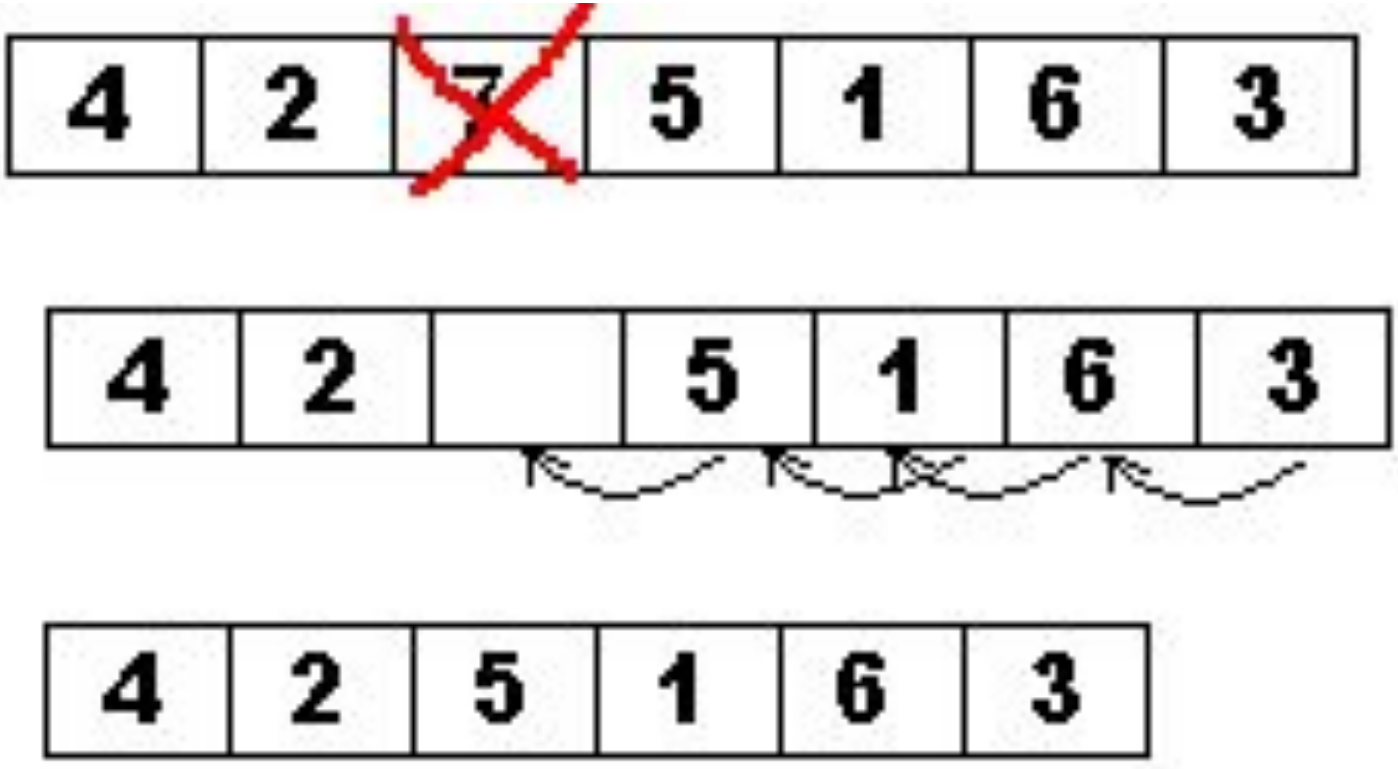
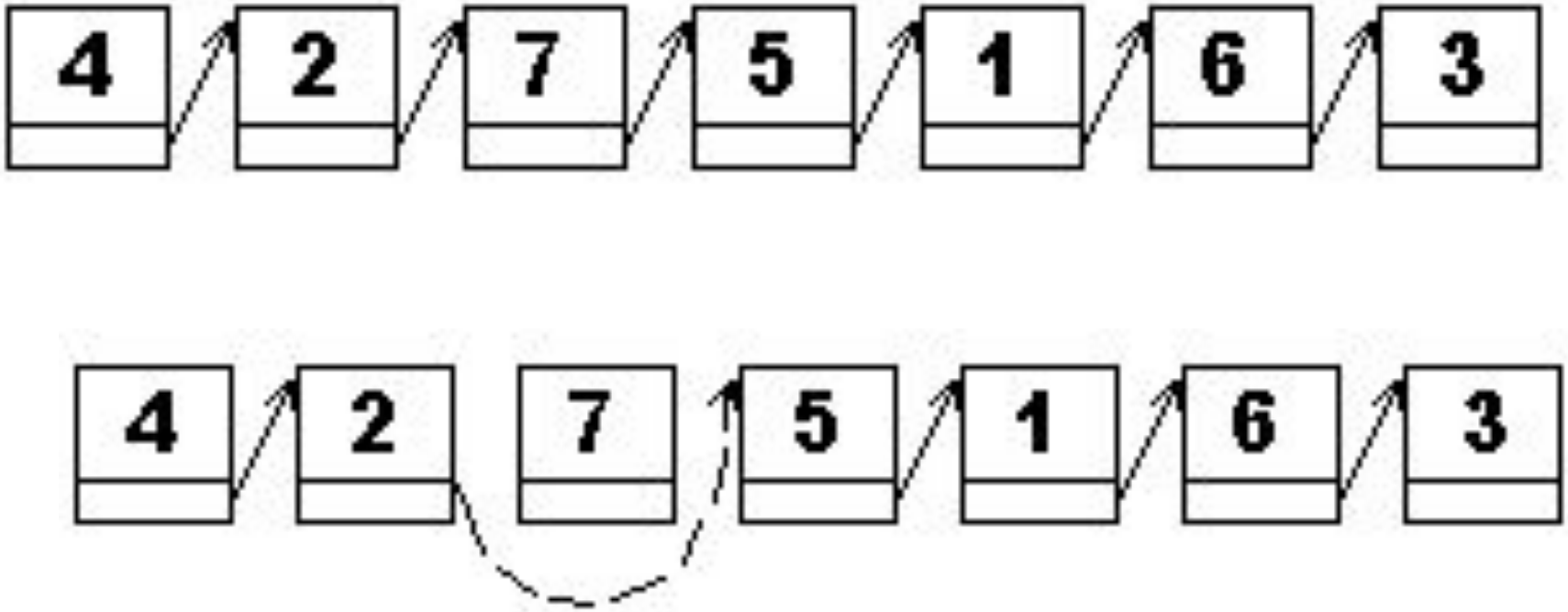
Реализация стека в виде массива. Функция исключения элемента из стека

```
1.  void iscluch ( *int y)
2.   {
3.      if( TOP==0 )
4.       {   printf(“Нехватка элементов”);
5.           return;  }
6.    *y = Stack[TOP];
7.     TOP--;
8.  }
```

Реализация стека в виде массива

```
1.   int main()
2.   {
3.      int y, i;
4.      for (i=0; i<10; i++)
5.      {
6.         scanf(“%d”, &y);
7.          vcluch (y);
8.      }
9.      iscluch(&y);
10.     printf(“%d”, y);
11.   }
```

Линейный список в виде одномерного массива



Линейный список в динамической памяти