Потоци и файлове

(преговор)

Трифон Трифонов

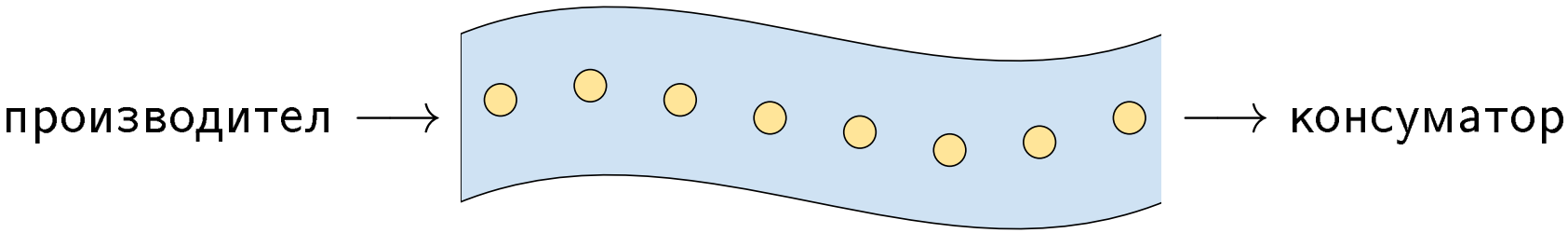
Структури от данни и програмиране, спец. Компютърни науки, 2 поток, 2024/25 г.

17 октомври 2017 г.

"Blue Stream" от Fitz Gerard Villafuerte (https://flic.kr/p/7FmKBb), CC BY-NC-ND 2.0

"Hanging Files" от Reeding Lessons (https://flic.kr/p/7ihBVd), CC BY-NC-SA 2.0

Абстракцията поток



Поточен буфер

- Какво представлява буферът?

Поточен буфер

- Какво представлява буферът?
- Кога е нужен буфер?

Поточен буфер

- Какво представлява буферът?
- Кога е нужен буфер?
- Кога буферът вреди?

Поточен буфер

- Какво представлява буферът?
- Кога е нужен буфер?
- Кога буферът вреди?

H	e	l	l	o	,		w	o	r	l	d	!	\n

Стандартни потоци и пренасочване

- Стандартен изходен поток `cout` (`stdout`)
  - Пренасочване на изхода:
  - `ls > filelist.txt`

Стандартни потоци и пренасочване

- Стандартен изходен поток `cout` (`stdout`)
  - Пренасочване на изхода:
  - `ls > filelist.txt`
- Стандартен входен поток `cin` (`stdin`)
  - Пренасочване на вход и на изход:
  - `grep password < email.txt > password.txt`

Стандартни потоци и пренасочване

- Стандартен изходен поток `cout` (`stdout`)
  - Пренасочване на изхода:
  - `ls > filelist.txt`
- Стандартен входен поток `cin` (`stdin`)
  - Пренасочване на вход и на изход:
  - `grep password < email.txt > password.txt`
- Стандартен поток за грешки `cerr` (`stderr`)
  - Пренасочване на изход за грешки:
  - `mv *.dat /data 2> errors.txt`

Стандартни потоци и пренасочване

- Стандартен изходен поток `cout` (`stdout`)
  - Пренасочване на изхода:
  - `ls > filelist.txt`
- Стандартен входен поток `cin` (`stdin`)
  - Пренасочване на вход и на изход:
  - `grep password < email.txt > password.txt`
- Стандартен поток за грешки `cerr` (`stderr`)
  - Пренасочване на изход за грешки:
  - `mv *.dat /data 2> errors.txt`
- Стандартен поток за дневник `clog` (отново `stderr`)

Форматиран и неформатиран вход/изход

- Текстова и двоична информация

Форматиран и неформатиран вход/изход

- Текстова и двоична информация
- ASCII (`char`)

Форматиран и неформатиран вход/изход

- Текстова и двоична информация
- ASCII (`char`)
- Служебни символи

Форматиран и неформатиран вход/изход

- Текстова и двоична информация
- ASCII (`char`)
- Служебни символи
- Кодиращи таблици

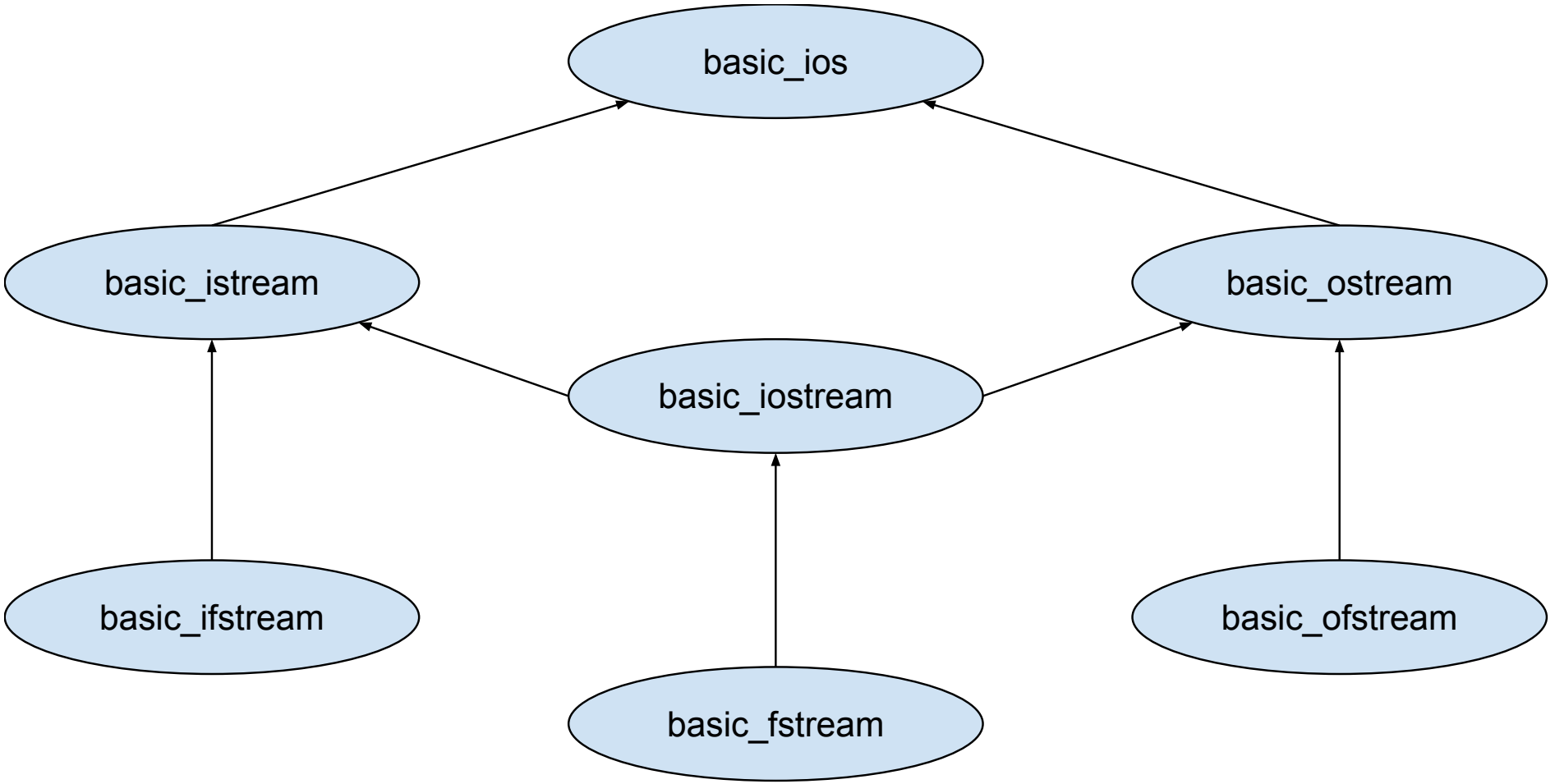
Форматиран и неформатиран вход/изход

- Текстова и двоична информация
- ASCII (`char`)
- Служебни символи
- Кодиращи таблици
- Unicode (`wchar_t`)

Форматиран и неформатиран вход/изход

- Текстова и двоична информация
- ASCII (`char`)
- Служебни символи
- Кодиращи таблици
- Unicode (`wchar_t`)
- UTF-8

Поточна йерархия в C++



Изход на поток

Неформатиран изход:

```
ostream& put(char);
ostream& write(const char*, streamsize);
```

Изход на поток

Неформатиран изход:

```
ostream& put(char);
ostream& write(const char*, streamsize);
```

Форматиран изход:

```
ostream& operator<<(ostream&, T);
```

Вход от поток

Неформатиран вход:

```
istream& get(char&);
istream& get(char*,streamsize,char);
istream& getline(char*,streamsize,char);
streamsize gcount() const;
istream& read(char*, streamsize);
```

Вход от поток

Неформатиран вход:

```
istream& get(char&);
istream& get(char*,streamsize,char);
istream& getline(char*,streamsize,char);
streamsize gcount() const;
istream& read(char*, streamsize);
```

Форматиран вход:

```
istream& operator>>(istream&, T&);
```

Вход от поток

Неформатиран вход:

```
istream& get(char&);
istream& get(char*,streamsize,char);
istream& getline(char*,streamsize,char);
streamsize gcount() const;
istream& read(char*, streamsize);
```

Форматиран вход:

```
istream& operator>>(istream&, T&);
```

Допълнителни функции:

```
int peek();
istream& putback(char);
```

Низови потоци

```
#include <sstream>
```

Входен поток от низ: `istringstream`

Пример:

```
char s[] = "1 2 3";
istringstream iss(s);
int a, b, c;
iss >> a >> b >> c;
```

Низови потоци

```
#include <sstream>
```

Входен поток от низ: `istringstream`

Пример:

```
char s[] = "1 2 3";
istringstream iss(s);
int a, b, c;
iss >> a >> b >> c;
```

Изходен поток към низ: `ostringstream`

Пример:

```
ostringstream oss;
oss << 1.2 << ' ' << 3.4;
cout << oss.str();
```

Състояние на поток

Флагове за състояние:

iostate	goodbit	eofbit	failbit	badbit
	0	1	2	4

Състояние на поток

Флагове за състояние:

iostate	goodbit	eofbit	failbit	badbit
	0	1	2	4

Селектори:

```
bool good() const; bool eof() const;
bool fail() const; bool bad() const;
iostate rdstate() const;
```

Мутатор:

```
void clear(iostate = 0);
```

Примери:

```
if (cin.rdstate() & (eofbit | badbit)) ...
cin.clear(failbit);
if(cin)...          if(!cin)...
```

Потокови манипулатори

```
#include<iomanip>
stream << data1 << manipulator << data2;
```

- Манипулатори за изход: `endl`, `ends`, `flush`
- Манипулатори за бройна система: `hex`, `oct`, `dec`
- Манипулатори за поле: `setw`, `setfill`, `left`, `right`, `internal`
- Манипулатори за дробни числа: `fixed`, `scientific`, `setprecision`
- Манипулатори за формат: `setiosflags`, `setbase`
- ...и много други

Какво е файл?

- Блок информация, записана на траен носител

Какво е файл?

- Блок информация, записана на траен носител
- Разлика между масив и файл

Какво е файл?

- Блок информация, записана на траен носител
- Разлика между масив и файл
- Файлови системи

Какво е файл?

- Блок информация, записана на траен носител
- Разлика между масив и файл
- Файлови системи
- Метаданни на файла

Файлове и потоци

Файлът като поток:

- Последователен достъп

Файлове и потоци

Файлът като поток:

- Последователен достъп
- Еднопосочно обхождане

Файлове и потоци

Файлът като поток:

- Последователен достъп
- Еднопосочно обхождане
- Еднократна обработка

Файлове и потоци

Файлът като поток:

- Последователен достъп
- Еднопосочно обхождане
- Еднократна обработка
- Краен поток

Файлове и потоци

Файлът като поток:

- Последователен достъп
- Еднопосочно обхождане
- Еднократна обработка
- Краен поток
- Файлът може да играе ролята на

Файлове и потоци

Файлът като поток:

- Последователен достъп
- Еднопосочно обхождане
- Еднократна обработка
- Краен поток
- Файлът може да играе ролята на
  - производител (входни файлове)

Файлове и потоци

Файлът като поток:

- Последователен достъп
- Еднопосочно обхождане
- Еднократна обработка
- Краен поток
- Файлът може да играе ролята на
  - производител (входни файлове)
  - консуматор (изходни файлове)

Файлове и потоци

Файлът като поток:

- Последователен достъп
- Еднопосочно обхождане
- Еднократна обработка
- Краен поток
- Файлът може да играе ролята на
  - производител (входни файлове)
  - консуматор (изходни файлове)

Файлът не е само поток:

Файлове и потоци

Файлът като поток:

- Последователен достъп
- Еднопосочно обхождане
- Еднократна обработка
- Краен поток
- Файлът може да играе ролята на
  - производител (входни файлове)
  - консуматор (изходни файлове)

Файлът не е само поток:

- Пряк достъп

Файлове и потоци

Файлът като поток:

- Последователен достъп
- Еднопосочно обхождане
- Еднократна обработка
- Краен поток
- Файлът може да играе ролята на
  - производител (входни файлове)
  - консуматор (изходни файлове)

Файлът не е само поток:

- Пряк достъп
- Разширяване при запис

Файлове и потоци

Файлът като поток:

- Последователен достъп
- Еднопосочно обхождане
- Еднократна обработка
- Краен поток
- Файлът може да играе ролята на
  - производител (входни файлове)
  - консуматор (изходни файлове)

Файлът не е само поток:

- Пряк достъп
- Разширяване при запис
- Едновременно четене и запис

Текстови и двоични файлове

Текстови файлове

- Форматиран вход и изход

Текстови и двоични файлове

Текстови файлове

- Форматиран вход и изход
- Само последователен достъп

Текстови и двоични файлове

Текстови файлове

- Форматиран вход и изход
- Само последователен достъп
- Еднократно обхождане

Текстови и двоични файлове

Текстови файлове

- Форматиран вход и изход
- Само последователен достъп
- Еднократно обхождане
- Интерпретиране на данните във файла като текст (ASCII, Unicode или др.)

Текстови и двоични файлове

Текстови файлове

- Форматиран вход и изход
- Само последователен достъп
- Еднократно обхождане
- Интерпретиране на данните във файла като текст (ASCII, Unicode или др.)
- Прилича на низ

Текстови и двоични файлове

Текстови файлове

- Форматиран вход и изход
- Само последователен достъп
- Еднократно обхождане
- Интерпретиране на данните във файла като текст (ASCII, Unicode или др.)
- Прилича на низ

Двоични файлове

Текстови и двоични файлове

Текстови файлове

- Форматиран вход и изход
- Само последователен достъп
- Еднократно обхождане
- Интерпретиране на данните във файла като текст (ASCII, Unicode или др.)
- Прилича на низ

Двоични файлове

- Неформатиран (суров) вход и изход

Текстови и двоични файлове

Текстови файлове

- Форматиран вход и изход
- Само последователен достъп
- Еднократно обхождане
- Интерпретиране на данните във файла като текст (ASCII, Unicode или др.)
- Прилича на низ

Двоични файлове

- Неформатиран (суров) вход и изход
- Позволява пряк достъп

Текстови и двоични файлове

Текстови файлове

- Форматиран вход и изход
- Само последователен достъп
- Еднократно обхождане
- Интерпретиране на данните във файла като текст (ASCII, Unicode или др.)
- Прилича на низ

Двоични файлове

- Неформатиран (суров) вход и изход
- Позволява пряк достъп
- Многократно обхождане

Текстови и двоични файлове

Текстови файлове

- Форматиран вход и изход
- Само последователен достъп
- Еднократно обхождане
- Интерпретиране на данните във файла като текст (ASCII, Unicode или др.)
- Прилича на низ

Двоични файлове

- Неформатиран (суров) вход и изход
- Позволява пряк достъп
- Многократно обхождане
- Интерпретацията на данните във файла зависи от конкретната задача

Текстови и двоични файлове

Текстови файлове

- Форматиран вход и изход
- Само последователен достъп
- Еднократно обхождане
- Интерпретиране на данните във файла като текст (ASCII, Unicode или др.)
- Прилича на низ

Двоични файлове

- Неформатиран (суров) вход и изход
- Позволява пряк достъп
- Многократно обхождане
- Интерпретацията на данните във файла зависи от конкретната задача
  - масив от числа

Текстови и двоични файлове

Текстови файлове

- Форматиран вход и изход
- Само последователен достъп
- Еднократно обхождане
- Интерпретиране на данните във файла като текст (ASCII, Unicode или др.)
- Прилича на низ

Двоични файлове

- Неформатиран (суров) вход и изход
- Позволява пряк достъп
- Многократно обхождане
- Интерпретацията на данните във файла зависи от конкретната задача
  - масив от числа
  - структура

Текстови и двоични файлове

Текстови файлове

- Форматиран вход и изход
- Само последователен достъп
- Еднократно обхождане
- Интерпретиране на данните във файла като текст (ASCII, Unicode или др.)
- Прилича на низ

Двоични файлове

- Неформатиран (суров) вход и изход
- Позволява пряк достъп
- Многократно обхождане
- Интерпретацията на данните във файла зависи от конкретната задача
  - масив от числа
  - структура
  - масив от структури

Входни файлове

```
ifstream(char const*, openmode = ios::in )
```

- `void open(char const*, openmode = ios::in)`
- `void close()`
- `ios::binary` — суров (неформатиран) вход

Входни файлове

```
ifstream(char const*, openmode = ios::in )
```

- `void open(char const*, openmode = ios::in)`
- `void close()`
- `ios::binary` — суров (неформатиран) вход

Примери:

```
ifstream fi("email.txt", ios::in );
ifstream fi("lolcat.jpg", ios::in | ios::binary );
```

Изходни файлове

```
ofstream(char const*, openmode = ios::out|ios::trunc)
```

- `void open(char const*, openmode)`
- `void close()`
- `ios::trunc` — отрязва (унищожава) файла
- `ios::ate` — вмъкването става в края
- `ios::app` — вмъкването винаги е в края

Изходни файлове

```
ofstream(char const*, openmode = ios::out|ios::trunc)
```

- `void open(char const*, openmode)`
- `void close()`
- `ios::trunc` — отрязва (унищожава) файла
- `ios::ate` — вмъкването става в края
- `ios::app` — вмъкването винаги е в края

Примери:

```
ofstream fo("page.html", ios::out );
ofstream fo("application.log", ios::out | ios::app );
ofstream fo("file.dat", ios::out | ios::binary );
```

Входно-изходни файлове

```
fstream(char const*, openmode = ios::in | ios::out)
```

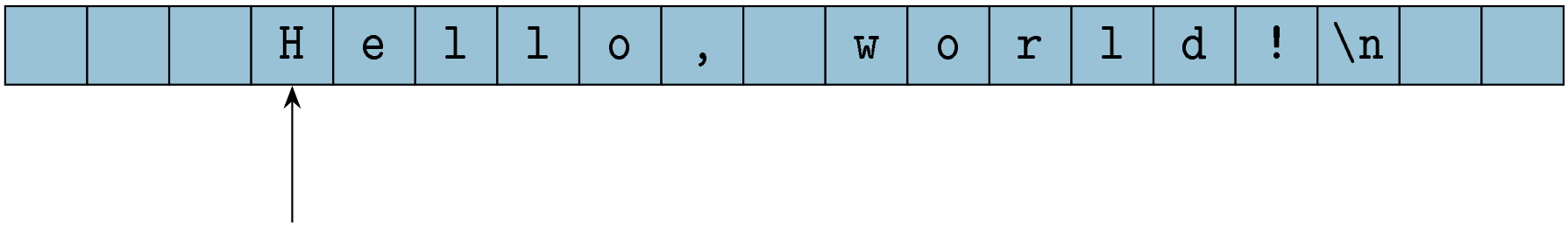
Входно-изходни файлове

```
fstream(char const*, openmode = ios::in | ios::out)
```

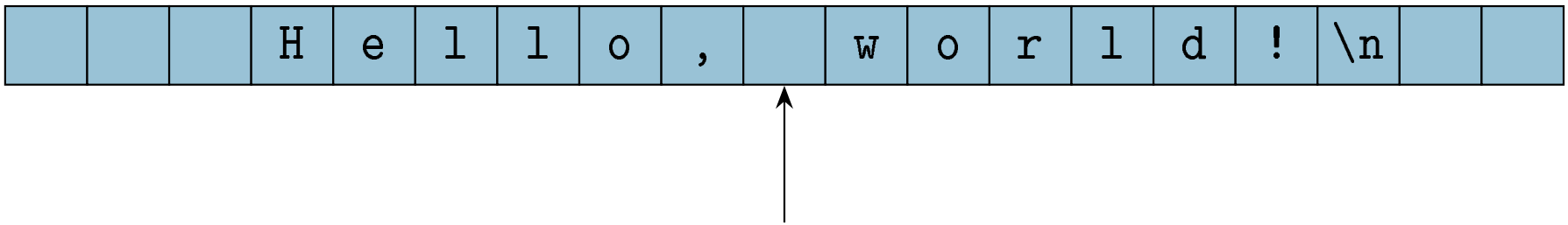
Пример:

```
fstream f( "essay.txt" );
f.getline(line, 100);
f << "Ignore the following text, please!";
```

Файлов указател



Файлов указател



Пряк достъп до файлове

Отправна точка за преместване на текущата позиция:

seekdir	beg	cur	end

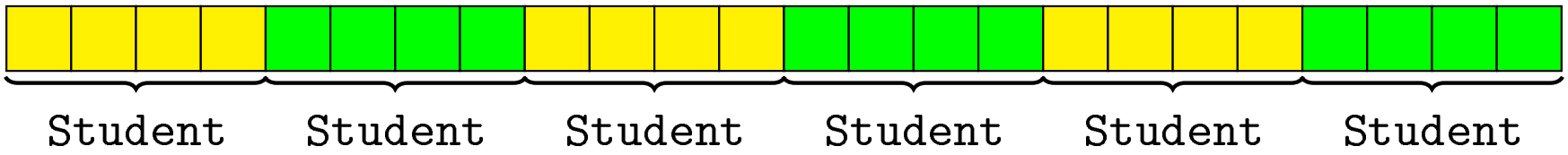
Селектори:

```
streampos tellg() const
streampos tellp() const
```

Мутатори:

```
istream& seekg(streampos, seekdir = beg)
ostream& seekp(streampos, seekdir = beg)
```

Блокова организация




```
class Student { ... };

Student s;
f.seekp( i * sizeof (Student) );
f.write((char const*)&s, sizeof(Student));

Student sa[3];
f.seekg( j * sizeof(Student) );
f.read( (char*)sa, 3 * sizeof(Student));
```

Задача "СУСИ"

- Да се въведе списък от студенти
- Да се запише в текстов файл `students.txt`
- От `students.txt` да се прочетат студентите, които не са скъсани и да се запишат в главната книга `main.bk`
- В главната книга да се повиши с 1.0 оценката на студент с даден Ф№