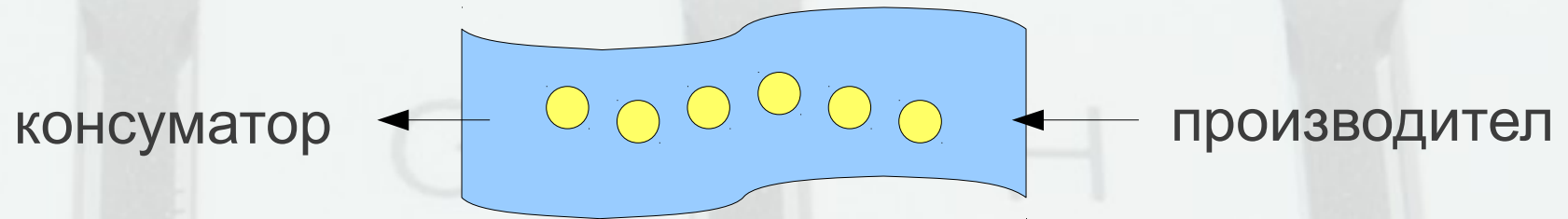


ПОТОЦИ



Абстракцията поток



Обектно-ориентиран подход

- `cin >> number >> char >> string;`
- `file << student << list << tree_of_files;`
- `while (stream1 >> x) stream2 << f(x);`

Поточна обработка

- Конвейер
- Unix pipes
ls | grep new | wc -l
- Позволява паралелна обработка
- Файловете като потоци

Буфериране

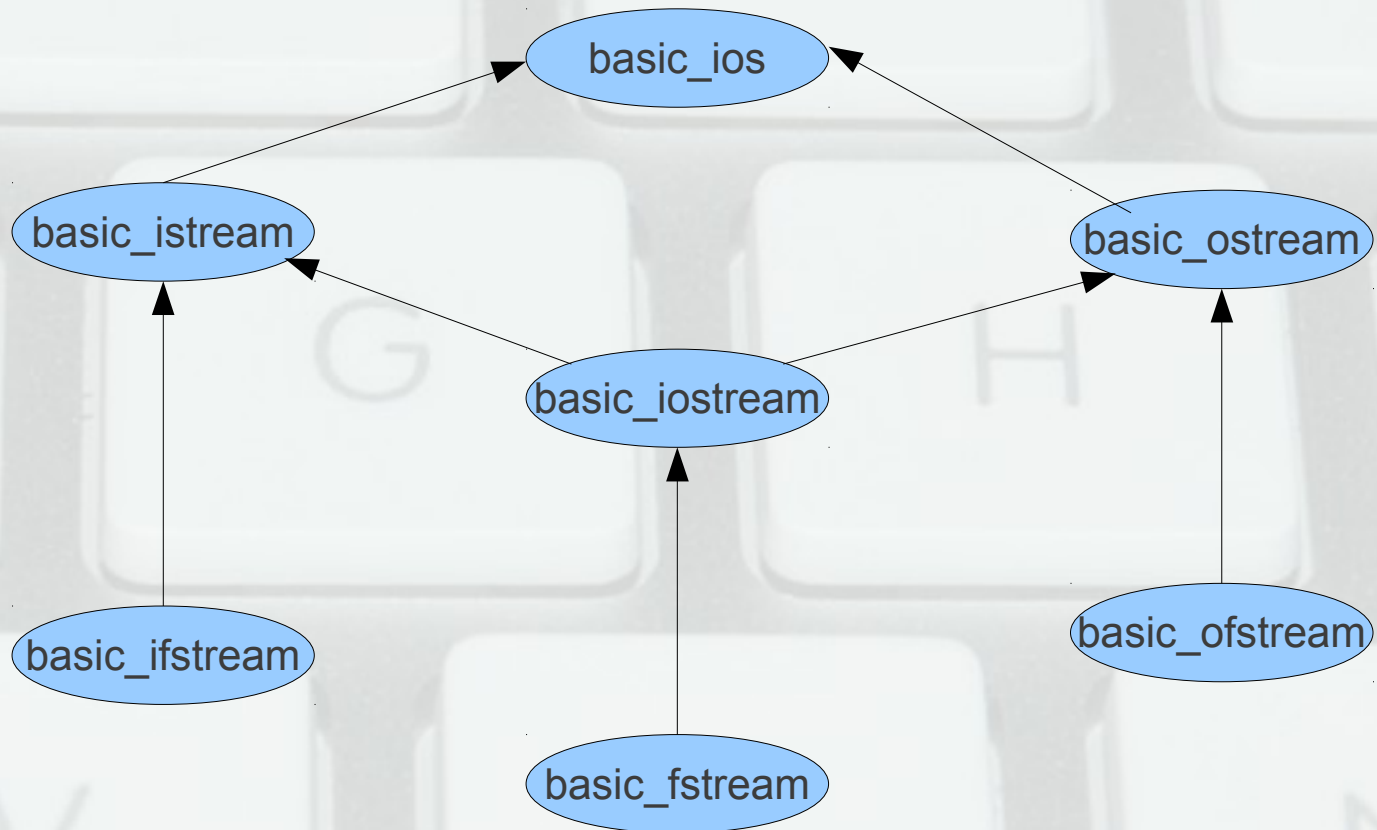
- Какво е?
- Защо има нужда от него?
- Кога няма нужда от него?

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

Стандартни потоци и пренасочване

- `cout (stdout)`
 - `dir > filelist.txt`
- `cin (stdin)`
 - `grep password < email.txt > password.txt`
- `cerr (stderr)`
 - `move *.dat d:\ 2> errors.txt`
- `clog (отново stderr)`

Поточна йерархія



Форматиран и неформатиран ВХОД/ИЗХОД

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

Функции за ИЗХОД

- `ostream& put(char);`
- `ostream& write(const char*, streamsize);`
- `ostream& operator<<(ostream&, T);`

Функции за ВХОД

- `istream& get(char&);`
- `istream& get(char*, streamsize, char);`
- `istream& getline(char*, streamsize, char);`
- `streamsize gcount() const;`
- `istream& read(char*, streamsize);`
- `istream& operator>>(istream&, T);`

Още функции за вход

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

Операции << и >>

- `cin >> a >> b >> c;`
- `>>` и `<<` не може да са член-функции!
 - защо?
- *friend* `ostream& operator<<(ostream&, A const&);`
- *friend* `istream& operator>>(istream&, A&);`
- Предефиниране на операторите за вход и изход

СЪСТОЯНИЕ НА ПОТОК

- enum iostate { goodbit = 0, eofbit = 1,
failbit = 2, badbit = 4 }
- bool good() const;
- bool eof() const;
- bool fail() const;
- bool bad() const;

СЪСТОЯНИЕ НА ПОТОК

- `enum iostate { goodbit = 0, eofbit = 1, failbit = 2, badbit = 4 }`
- `iostate rdstate() const;`
- `void clear(iostate = 0);`
- `operator void*() const;`
- `bool operator !() const;`

Манипулатори или функции?

- `cout << setprecision(5)`
`cout.precision(5)`
- `cout << setw(10)`
`cout.width(10)`
- `cout << setfill('x')`
`cout.fill('x')`

Флагове за формат

- hex, oct, dec
- fixed, scientific
- left, right, internal
- `fmtflags flags() const;`
- `setf(fmtflags flg);`
- `stream << setiosflags(fmtflags);`

Манипулатори

- `stream << data1 << manipulator << data2;`
- `cout << 20 << hex << 20 << oct << 20;`
- `cout << scientific << 100 << fixed << 100;`

Потребителски манипулатори

- `istream& (*)(istream&);`
- `ostream& (*)(ostream&);`

`stream >> manipulator >> data`
↕
`manipulator(stream) >> data`

`stream << manipulator << data`
↕
`manipulator(stream) << data`