# Стандартна библиотека на C++ (std)

# Какво съдържа стандартната библиотека?

- Помощни функции, типове и дефиниции
- Реализация на често използвани класове
- Работа с вход и изход
- Шаблонни реализации на популярни структури от данни и алгоритми

# Досега сме говорили за...

- #include <iostream> — вход и изход
- #include <cmath> — математически функции
- #include <cstring> — функции за работа с низове
- #include <typeinfo> — дефинира type_info, връщан от typeid
- #include <exception> — дефинира стандартния клас exception
- Сега ще разгледаме някои други популярни части от std

# #include <cstddef>

- дефинира NULL ☺
- (и някои други неща)

# #include <cassert>

- assert(<израз>)
- проверява дали <израз> е истина, ако не — прекратява изпълнението програмата

# Граници на типовете

- #include <climits> — константи за граници на типовете
  - INT_MIN, INT_MAX, LONG_MIN, LONG_MAX, ...
- #include <cfloat> — точност на дробните типове
  - DBL_EPSILON, DBL_DIG, FLT_MAX_EXP
- #include <limits>
  - шаблон numeric_limits<T> със статични функции
  - Примери:
    - cout << numeric_limits<int>.min();
    - cout << numeric_limits<double>.max_exponent();

# #include <cstdlib>

- Разнообразни помощни функции
- exit(int c) — развива стека и прекратява програмата с код за грешка c
- abort() — прекратява програмата незабавно
- rand() — връща случайно цяло число
- конвертиращи функции
  - от низ към число — atoi, atof,...
  - от число към низ — strtoi, strtof, ...
- qsort — сортира елементи в масив

# #include <cctype>

- функции за работа със символи
  - проверка за вид: isalpha, isnum, isblank, ...
  - преобразуване: toupper, tolower
- работи с ASCII таблицата
- за работа с други азбуки и формати се ползват
  - #include<locale>
  - #include<clocale>
  - #include<cuchar>
  - #include<cwchar>
  - #include<cwctype>

# #include <cstdarg>

- работа с функции с произволен брой аргументи
- void printAll(int n, ...) {
  va_list vl;
  va_start(vl);
  for(int i = 0; i < n; i++)
    va_arg(vl, Printable*)->print();
  va_end(vl);
  }
- printAll(3, new Student, new Employee, new QuickTask);

# #include <ctime>

- Функции за работа с времена и дати
  - структура tm с полета за секунда, минута, ..., година
  - тип time_t описващ момент във времето
    - обикновено int: брой секунди изминали от 1.01.1970 г.
    - проблем 2038
  - time_t time(time_t* timer) — връща текущото време и го записва в timer, ако не е NULL
  - double difftime(time_t, time_t) — разлика в секунди между две времена
  - time_t mktime(tm*), tm* gmtime(time_t) — от tm в time_t и обратно
  - char* asctime(tm*), char* ctime(time_t) — от tm/time_t в низ

# #include <utility> — операции за сравнение

- пространство от имена rel_ops
- дефинира !=, <=, >=, > въз основа на == и <
- class Matrix {
  bool operator ==(Matrix const&) const;
  bool operator <(Matrix const&) const;
  };
- using namespace std::rel_ops;
- Matrix m1, m2; if (m1 >= m2) ...
- Недостатък: разпростира се за всички дефинирани класове!

# #include <utility> — шаблон за двойки

- template <typename T1, typename T2> struct pair {
  T1 first; T2 second; };
- pair<int, double> x(1, 3.5);
- pair<pair<int,double>,Task*> y(x, new QuickTask(„Тест“));
- cout << x.second << y.first.first;
- make_pair — шаблон на функция, позволява конструиране на двойка без да се задават типове
- make_pair(Student(40000, „Иван“, 5.30),
  SimpleTask(„Изпит по ООП“, 2));

# #include <string>

- string дефинира хубав клас за низ
- има голяма четворка и сам управлява динамичната си памет
- увеличава заделената памет при нужда
- конструктори
  - string(char const*) — от низ
  - string(char const*, int) — от масив от символи
  - string(int, char) — от повтарящ се символ
- c_str(), data() — преобразува до низ или масив от символи
- size(), length() — дължина на низа
- operator[](int), at(int) — връща i-ти символ (at проверява за коректност)

# #include <string> — сравнение и търсене

- с S ще отбелязваме string const&, char или char const*
- операции ==, !=, <, >, <=, >=
- compare(S) (връща -1, 0, 1 като strcmp)
- find(S, int=0)
  - търси първо срещане подниз или символ
  - започва от зададената позиция
  - връща npos, ако не е намерен
  - rfind търси последно срещане
- substr(int p, int n) — извлича подниз от позиция p с дължина n
- find_first_of(S), find_last_of(S), find_first_not_of(S), find_last_not_of(S)
  - търси първия/последния символ който се/не се среща в друг низ

# #include <string> — операции

- предефинирани са операциите <<, >>, getline
- operator+(S) слепва два низа и връща нов низ
- operator+=(S), append(S) залепват друг низ за дадения
- operator=(S), assign(S) заменят съдържанието на дадения низ
- insert(int p, S s) вмъква s на позиция p
- erase(int p, int n) изтрива n символа от позиция p
- replace(int p, int n, S s) { erase(p, n); insert(p, s); }
- swap(string&) разменя съдържанието на два низа

# За допълнителна информация

- http://www.cplusplus.com/
- Библиотеките Boost са популярно разширение на std
  - математика (алгебра, анализ, статистика, ...)
  - паралелно (конкурентно, многонишково) и мрежово програмиране
  - обработка на изображения
  - регулярни изрази
  - умни указатели
  - инструменти за създаване на интерпретатори и компилатори
  - и още много...
  - http://boost.org/