Тема 3-1.

Функциональная декомпозиция

для АСУб и ЭВМб

Темы лекции

- Модульное программирование и функциональная декомпозиция
- Использование функций в C++
- Особенности передачи параметров
- Сквозной пример II

Декомпозиция

Декомпозиция — операция мышления, состоящая в разделении целого на части.

Также декомпозицией называется общий приём, применяемый при решении проблем, состоящий в разделении проблемы на множество частных проблем, а также задач, с помощью объединения решений которых, можно сформировать решение исходной проблемы в целом.

Функциональная декомпозиция

1. Функциональная декомпозиция – это метод разработки программ, при котором задача разбивается на ряд легко решаемых подзадач, решения которых в совокупном виде дают решение исходной задачи в целом.

2. **Проектирование** приложения строится от абстрактного описания основной задачи (высший уровень абстракции). Основная задача может быть разбита на ряд более простых подзадач (второй уровень абстракции), каждая из которых, в свою очередь, на ряд еще более простых.

3. Процесс детализации заканчивается, когда очередная подзадача не может быть больше разбита на более простые составляющие, или когда решение очередной задачи становится очевидным.

Функциональная декомпозиция

1. В процессе детализации создается иерархические дерево решения, где каждый уровень представляет собой решение более детализированной задачи, чем предшествующий уровень.

2. Каждый блок представляет собой программный модуль. Каждый модуль, это законченный **алгоритм** решения некоторой конкретной задачи.

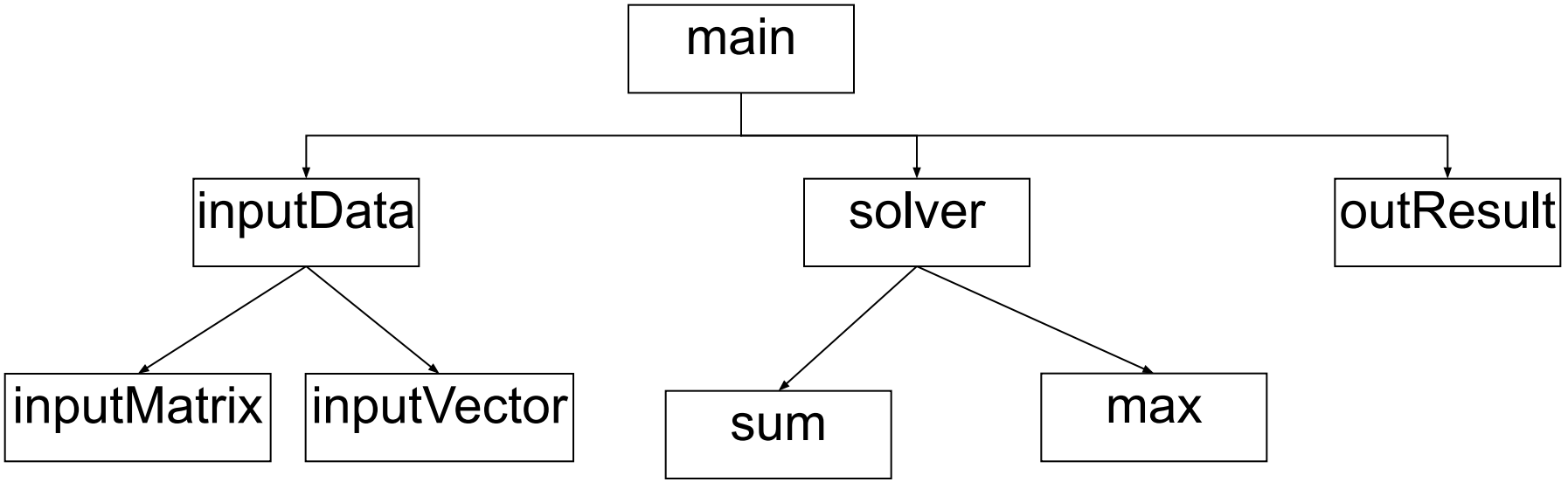
3. Процесс **кодирования** выполняется снизу вверх, от написания и полной отладки кода небольших подзадач с их последующей сборкой на верхнем уровне, при этом каждый модуль безошибочно решает одну задачу.

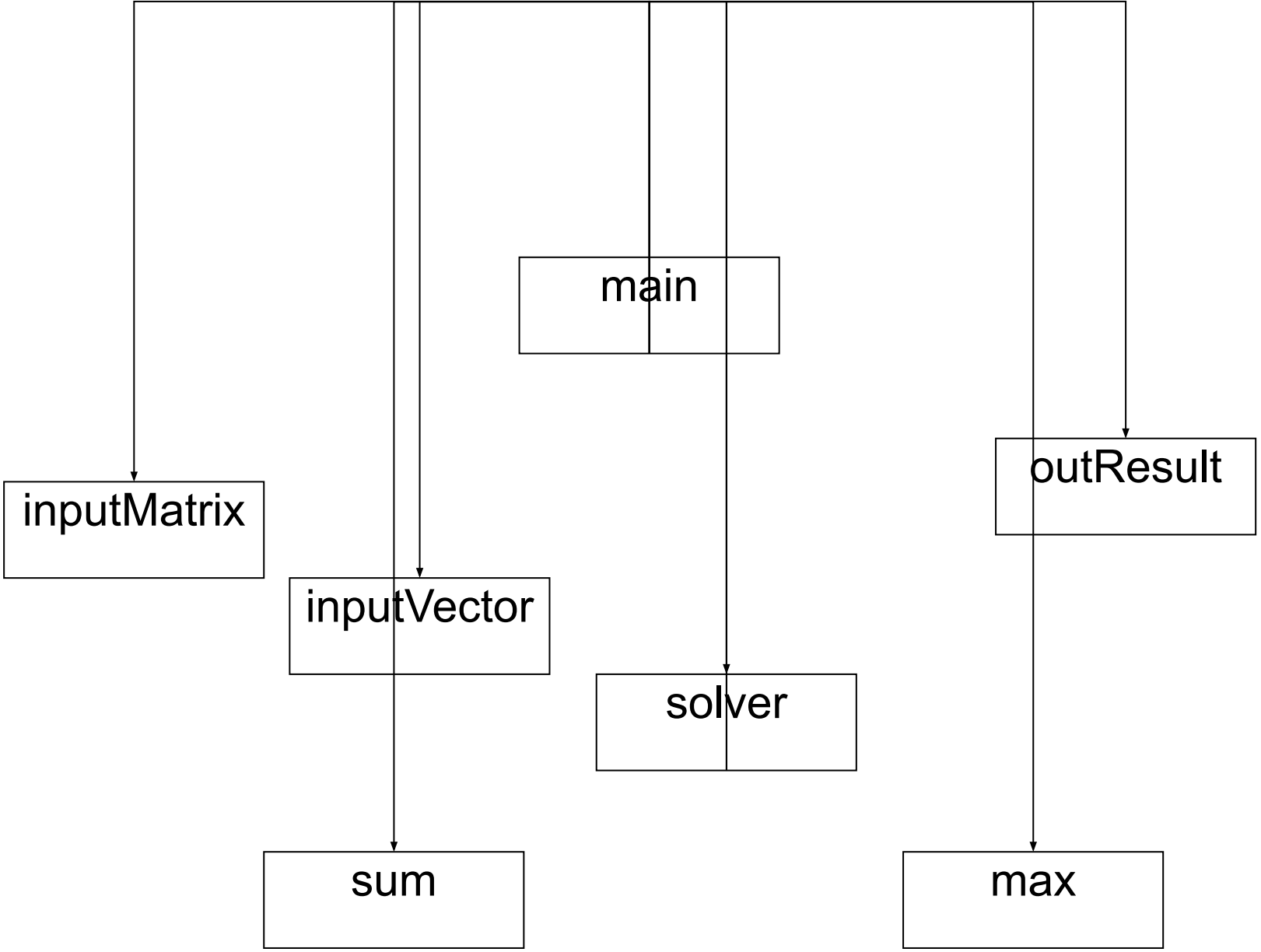
4. Объем задачи нижнего уровня достаточно небольшой, это одна или две страницы кода.

Пример задачи (из методички)

- Заданы две матрицы целых чисел A(m,n) и B(l,k) и два вектора из целых чисел X(n) и Y(k), где 0<m≤10 0<l≤10 0<n≤8 0<k≤8.
- Требуется получить вектора C(m) и D(l), где i=1..m i=1..l

$$C_i = \sum_{j=1}^{n} A_{i,j} X_j + \max A_{i,j} \qquad D_i = \sum_{j=1}^{k} B_{i,j} Y_j + \max B_{i,j}$$


main
inputData
solver
outResult
inputMatrix
inputVector
sum
max


main
inputMatrix
inputVector
solver
outResult
sum
max

Функции

Функции – относительно самостоятельные фрагменты программы, спроектированные для решения конкретных задач и снабженные именем.

Функция – это синтаксически выделенный именованный программный модуль, выполняющий определенное действие или группу действий.

Функции аналогичны программам в миниатюре и имеют общее название **подпрограммы**.

В объектно-ориентированных языках функции размещаются внутри классов и называются **методами**.

Преимущество функций

1. Экономия памяти кода за счет размещения многократно повторяющихся частей программ в функции.
2. Позволяет работать группе программистов над одной сложной задачей
3. Функции облегчают чтение, внесение изменений и коррекцию ошибок в программе.
4. Часто используемые функции помещают в библиотеки.

Основные понятия и определения

Каждая функция имеет свой интерфейс и реализацию.

Интерфейс функции – заголовок функции, в котором указывается название функции, список ее параметров и тип возвращаемого значения.

Реализация функции – тело функции, содержащее внутренние (локальные) данные функции и программный код, выполняющий действия согласно переданным в функцию параметрам и возвращающий значение, соответствующего интерфейсу функции типа.

Выводы о функциях (технический взгляд)

Способы реализации функций в разных языках программирования высокого уровня:

- объявления и функций
- правила именования функций
  - используемые символы, регистр
  - функции с одинаковыми названиями
- правила использования параметров функций
  - передача параметров
  - передача по порядку и по имени
  - значения по умолчанию

Выводы о функциях (методческий взгляд)

- Повторное использование кода
  - при повторении кода в программе объединять его в функцию
  - параметризация кода: минимизация использования конкретных значений (текст, числа и т.п.) и замена их на локальные переменные, входные параметры, значения по умолчанию
- Функциональная декомпозиция программы
  - Идеи для функций задаются типом его архитектуры (работа с устройствами ввода-вывода, корректность вводимых данных)
  - Идеи для функций задаются предметной логикой приложения (математическая библиотека, бухгалтерия, игры)
- Использование рекурсий
- Приёмы программирования