Computer Science I

CS 135

1. Introduction to Programming

René Doursat

Department of Computer Science & Engineering
University of Nevada, Reno

Fall 2005

Computer Science I

CS 135

0. Course Presentation
1. Introduction to Programming
2. Functions I: Passing by Value
3. File Input/Output
4. Predefined Functions
5. If and Switch Controls
6. While and For Loops
7. Functions II: Passing by Reference
8. 1-D and 2-D Arrays

Computer Science I

CS 135

1. Introduction to Programming

a. How to Develop a Program

b. Writing Pseudocode

c. First Elements of C++

d. Looking Under the Hood

Computer Science I

CS 135

1. Introduction to Programming

a. How to Develop a Program

- ✓ A program is like a recipe
- ✓ Steps in program development
- ✓ Procedural programming

b. Writing Pseudocode

c. First Elements of C++

d. Looking Under the Hood

1.a How to Develop a Program

A program is like a recipe

Pasta for six

- *boil 1 quart salty water*
- *stir in the pasta*
- *cook on medium until "al dente"*
- *serve*

1.a How to Develop a Program

A program is like a recipe

- What is programming?
  - ✓ programming can be defined as
    - the development of a solution to an identified problem, and
    - the setting up of a related series of instructions that will produce the desired results
  - ✓ generally, programming is the construction of an **algorithm**

1.a How to Develop a Program

A program is like a recipe

- What is an algorithm?
  - ✓ informally, a general method for solving a problem, such as a recipe
  - ✓ formally, a set of precise steps that describe exactly the tasks to be performed and in which order
  - ✓ an algorithm must
    - be precise and unambiguous
    - give the correct solution in all cases
    - eventually end
  - ✓ an algorithm frequently involves repetition of an operation

1.a How to Develop a Program

Steps in program development

- Seven basic steps in the development of a program

1. define the problem
2. outline the solution
3. develop the outline into an algorithm
4. test the algorithm for correctness
5. code the algorithm into a specific prog. language
6. run the program on the computer
7. document and maintain the program

1.a How to Develop a Program

Steps in program development

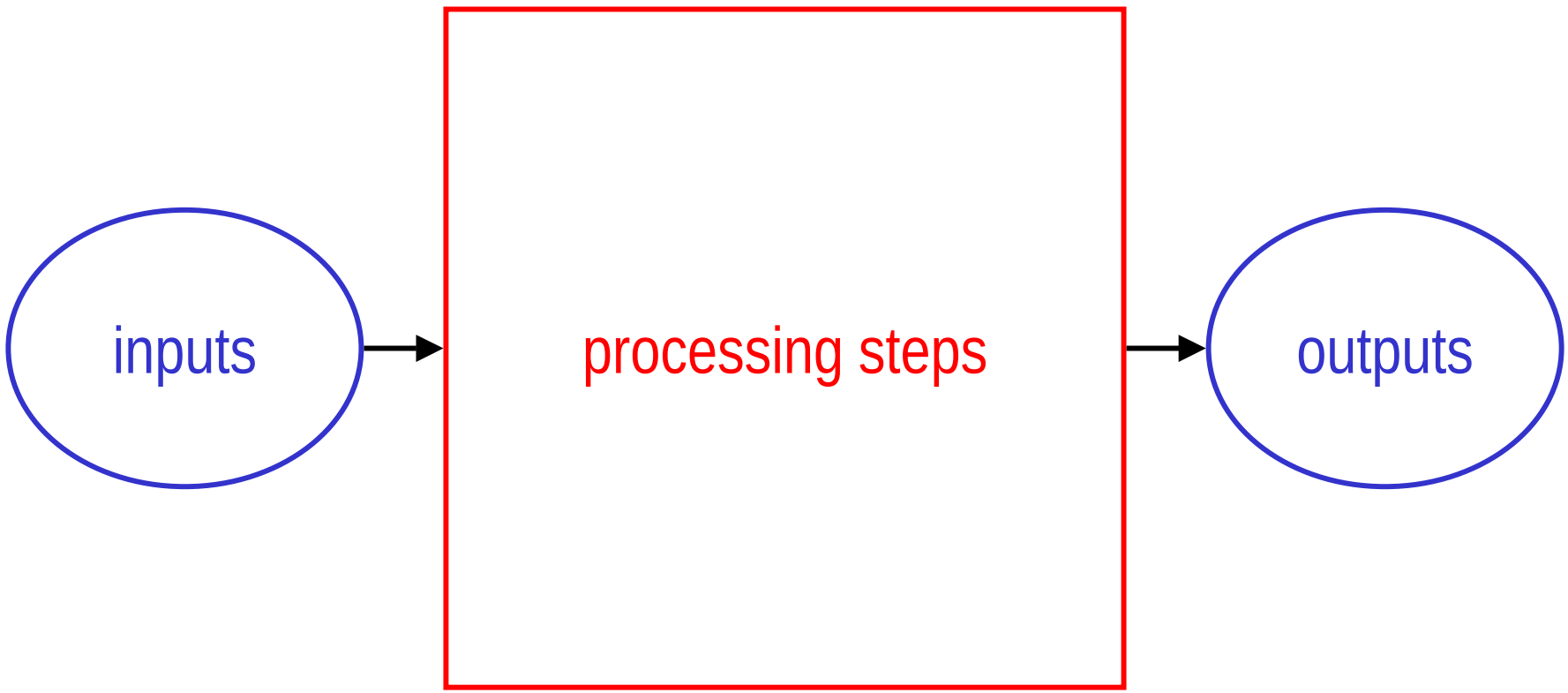
1. Define the problem

- ✓ to help with initial analysis, the problem should be divided into three separate components:
  - the inputs
  - the outputs
  - the processing steps to produce the required outputs from the inputs

1.a How to Develop a Program

Steps in program development

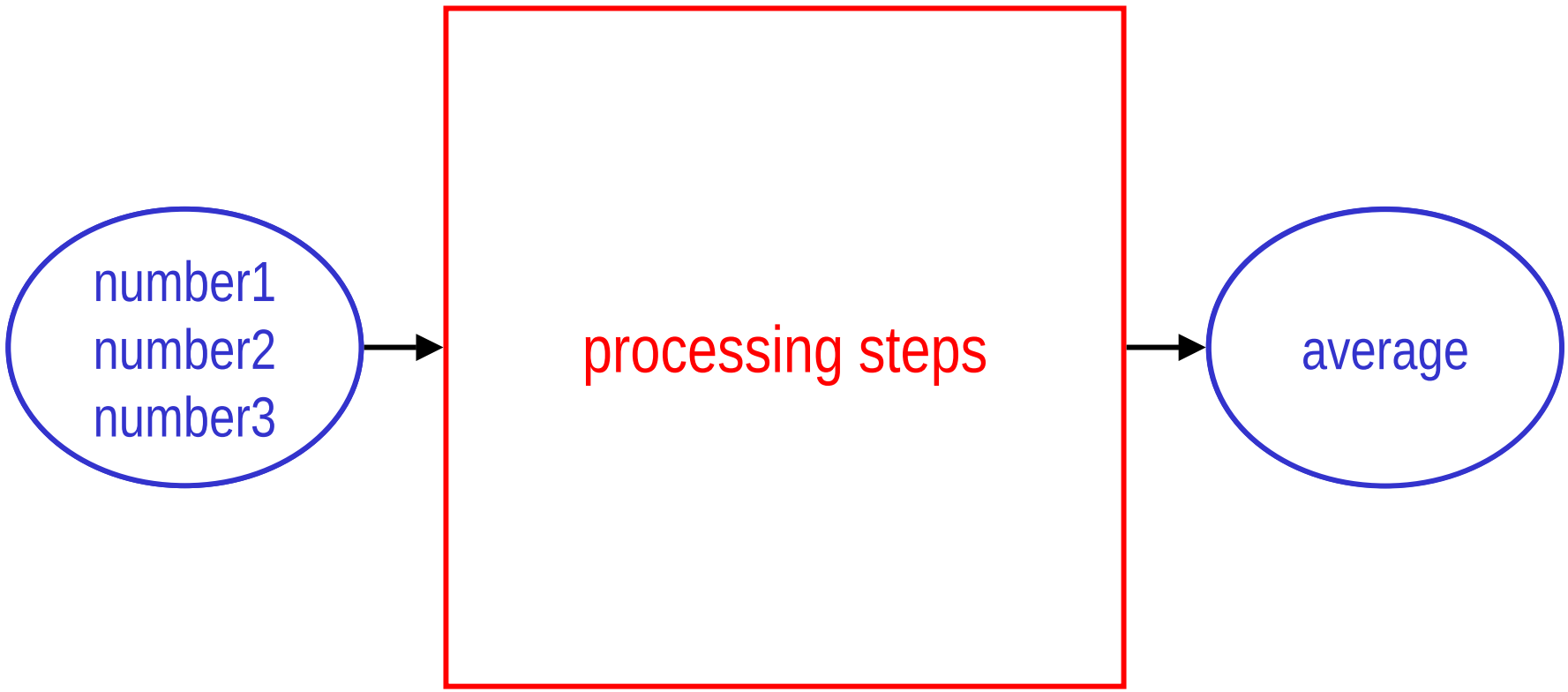
1. Define the problem



1.a How to Develop a Program

Steps in program development

- Example: find the average of three numbers
  - ✓ what are the inputs and outputs?



1.a How to Develop a Program

Steps in program development

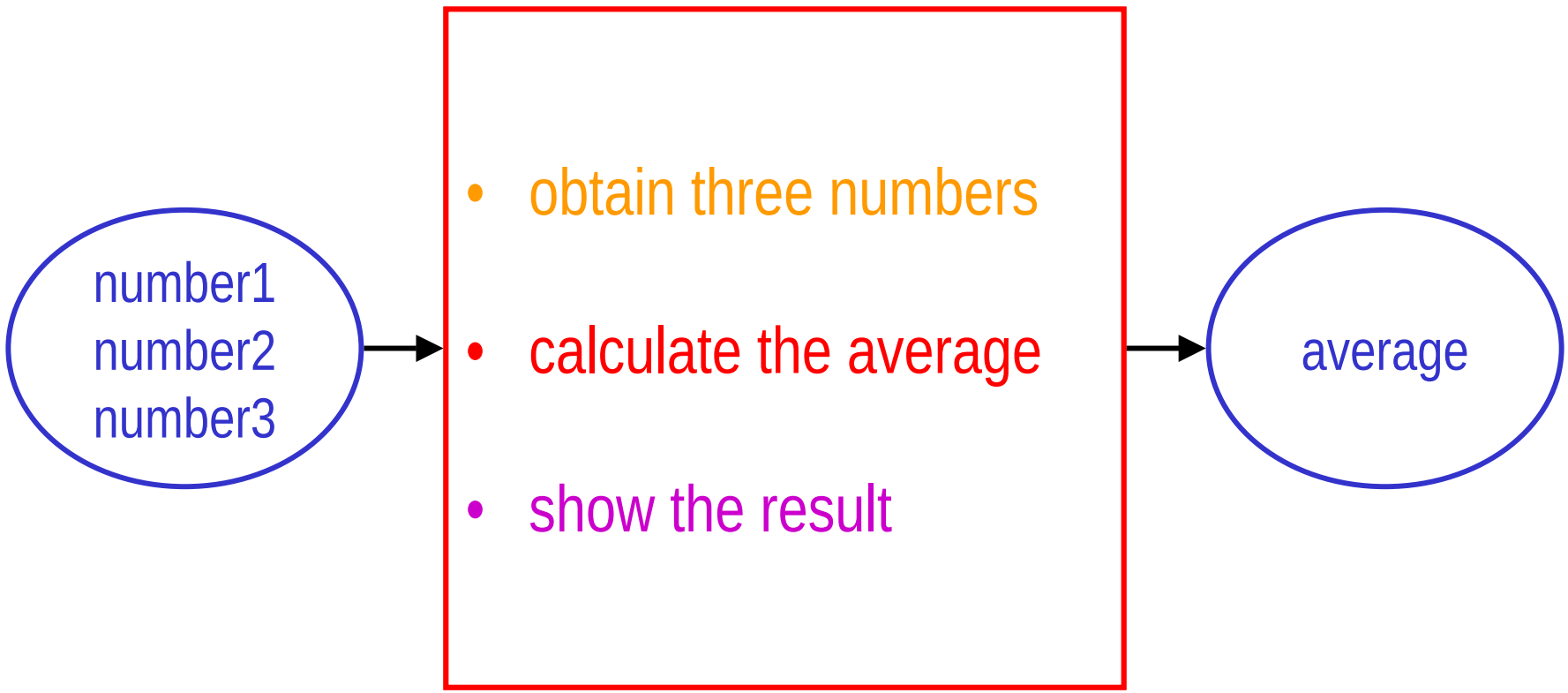
2. Outline the solution

- ✓ decompose the problem in smaller elements and produce a rough draft of the solution:
  - the major processing steps involved
  - the major subtasks (if any)
  - the major control structures
  - the major variables and record structures
  - the mainline logic

1.a How to Develop a Program

Steps in program development

- Example: find the average of three numbers
  - ✓ what are the processing steps?

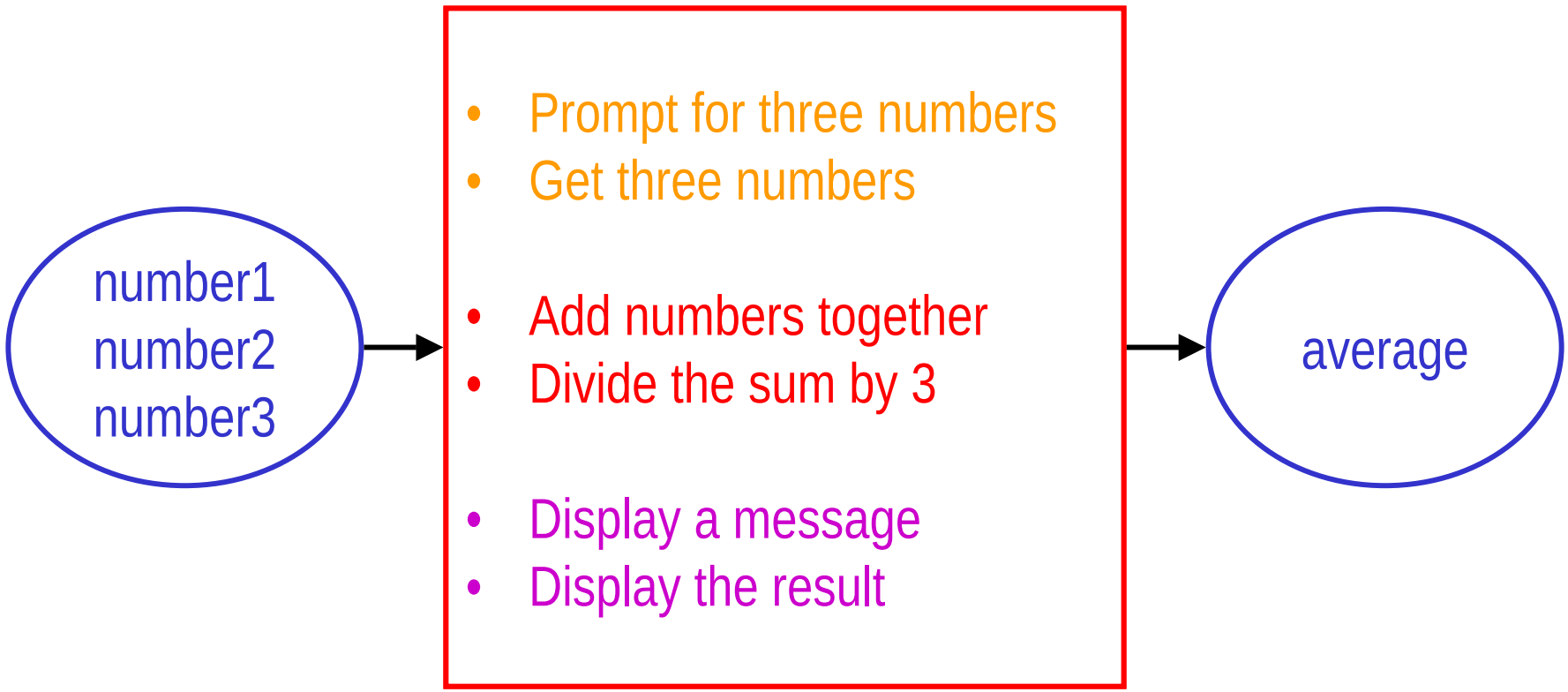


1.a How to Develop a Program

Steps in program development

3. Develop the outline into an algorithm

- ✓ the solution outline of Step 2 is expanded into an algorithm

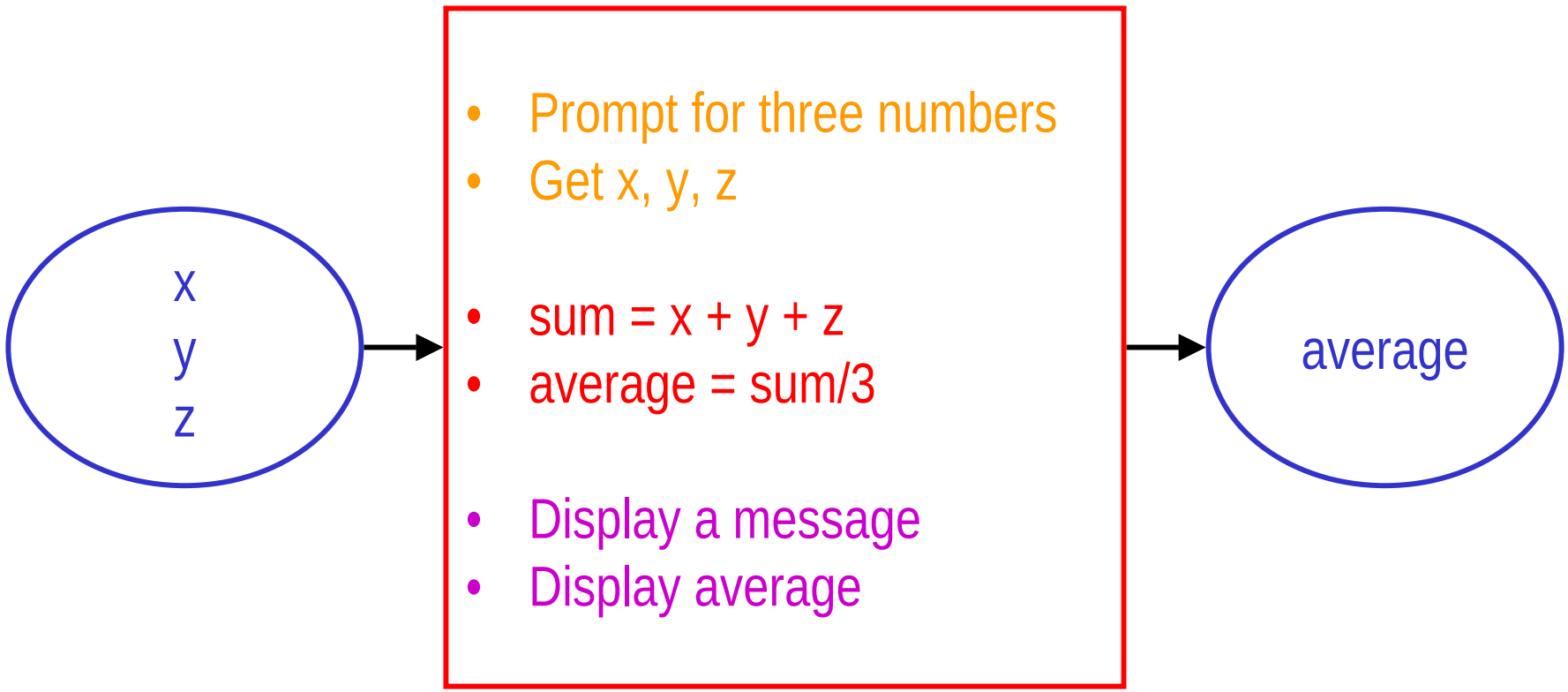


1.a How to Develop a Program

Steps in program development

3. Develop the outline into an algorithm

- ✓ here is an equivalent algorithm in a more formal style



1.a How to Develop a Program

Steps in program development

4. Test the algorithm for correctness

- ✓ testing is one of the most important step in the development of a program, yet it is often forgotten
- ✓ the main purpose of "desk-checking" the algorithm is to identify major logic errors early, so that they may be easily corrected

- sum = x + y + z
- average = sum/3

- ✓ try different test values by hand!

x	y	z	sum	avg
1	1	1	3	1
0	0	6	6	2
13	15	17	45	15
...	...	...	...	...

1.a How to Develop a Program

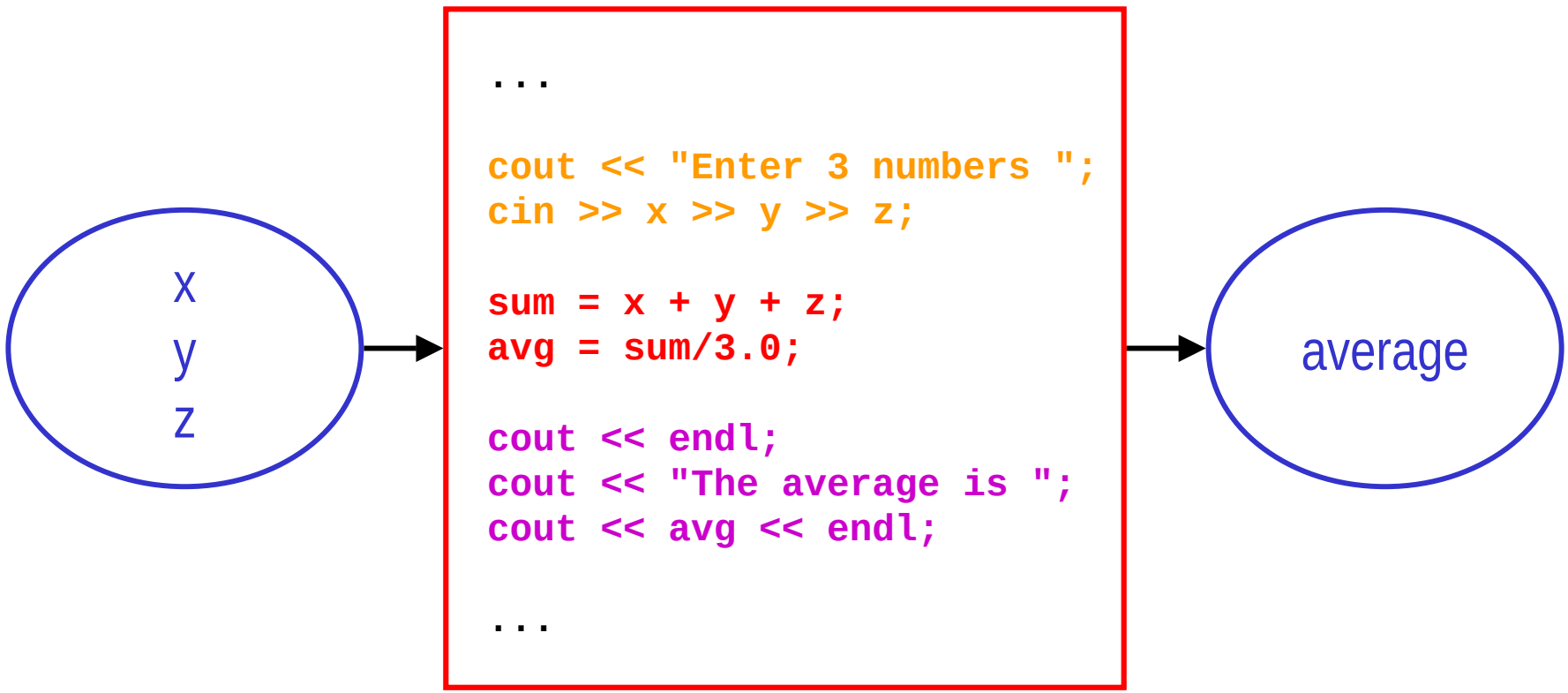
Steps in program development

5. Code the algorithm into a specific programming language
   - ✓ only after all design considerations have been met should you actually start to code the program into your chosen programming language:
     - C++
     - Java
     - FORTRAN
     - Basic
     - COBOL
     - etc.

1.a How to Develop a Program

Steps in program development

- Example: find the average of three numbers
  - ✓ core of the program in C++



1.a How to Develop a Program

Steps in program development

6. Run the program on the computer

- ✓ this step uses a program **compiler** and some test data to "machine-check" the code for errors:
  - syntax errors are detected at compile time
  - logic errors are detected at run time
- ✓ a compiler is like an interpreter: it translates a high-level language (such as C++) into low-level machine language (lots of 0's and 1's)
- ✓ compiling and running the program can be the most exciting and, at the same time, most frustrating part of the development process

1.a How to Develop a Program

Steps in program development

7. Document and maintain the program

- ✓ documentation should not be the last step but an ongoing task throughout the development process
  - external documentation: specifications, implementation, user manual, etc.
  - internal documentation: comments in the code

1.a How to Develop a Program

Steps in program development

- Summary

1. Define the problem
2. Outline the solution
3. Develop the outline into an algorithm
4. Test the algorithm for correctness

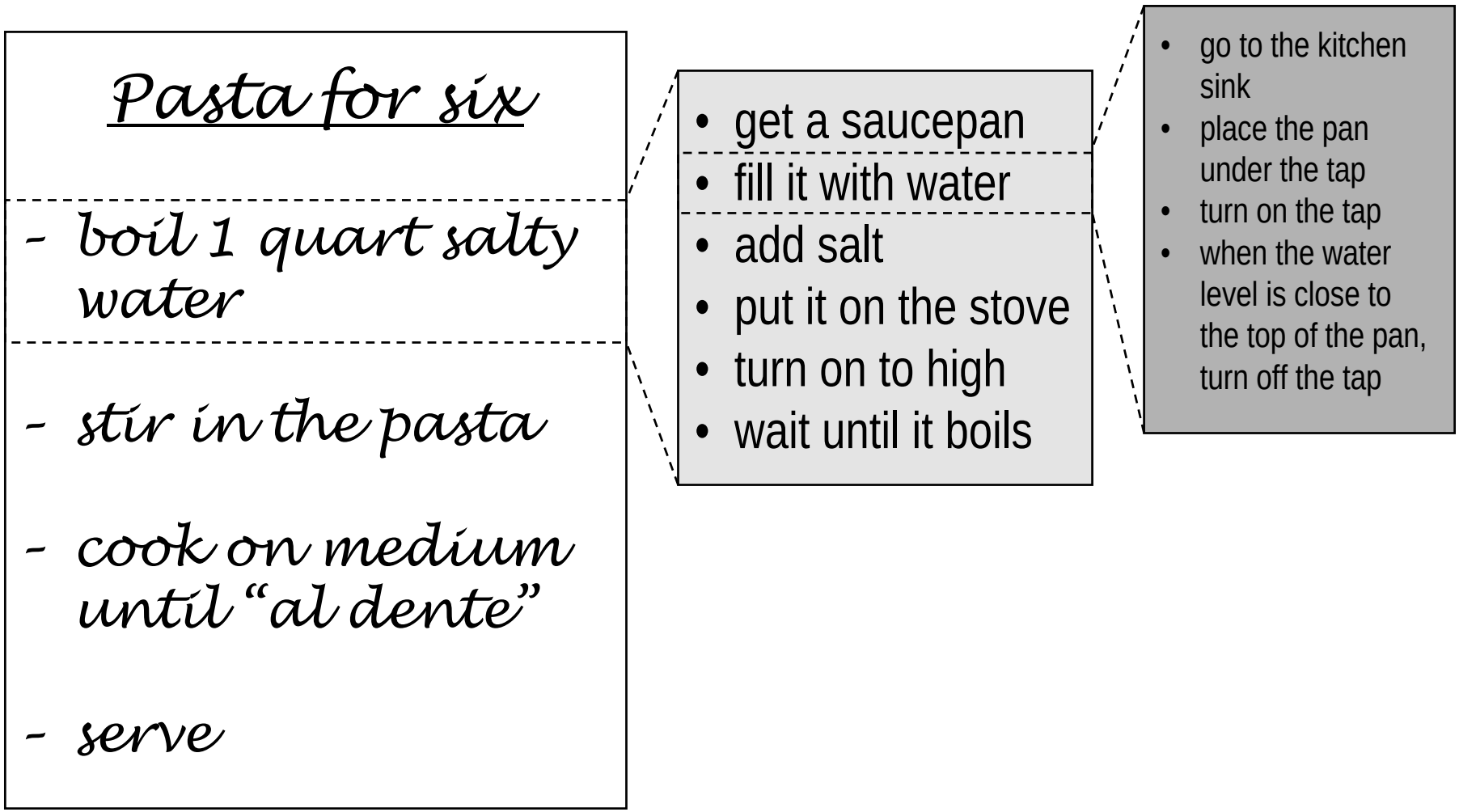
} design

5. Code the algorithm into a specific lang.
6. Run the program on the computer
7. Document and maintain the program

} implementation

1.a How to Develop a Program

Procedural programming



1.a How to Develop a Program

Procedural programming

- Top-down development
  - ✓ in the top-down development of a program design, a general solution to the problem is outlined first
  - ✓ this is then broken down gradually into more detailed steps until finally the most detailed levels have been completed
  - ✓ hierarchy of procedures, subtasks, and elementary steps
- Modular design
  - ✓ procedural programming also incorporates the concept of modular design, which involves grouping tasks together because they all perform the same function
  - ✓ modular design is connected directly to top-down development

Computer Science I

CS 135

1. Introduction to Programming

a. How to Develop a Program

- ✓ A program is like a recipe
- ✓ Steps in program development
- ✓ Procedural programming

b. Writing Pseudocode

c. First Elements of C++

d. Looking Under the Hood

Computer Science I

CS 135

1. Introduction to Programming

a. How to Develop a Program

b. Writing Pseudocode

- ✓ What is pseudocode?
- ✓ Six basic computer operations
- ✓ The structure theorem

c. First Elements of C++

d. Looking Under the Hood

1.b Writing Pseudocode

What is pseudocode?

Adding up a list of prices

Turn on calculator
Clear calculator
Repeat the following instructions
    Key in dollar amount
    Key in decimal point (.)
    Key in cents amount
    Press addition (+) key
Until all prices have been entered
Write down total price
Turn off calculator

1.b Writing Pseudocode

What is pseudocode?

- Pseudocode is a way to write an algorithm (recipe)
  - ✓ flowcharts are another popular way of representing algorithms
  - ✓ pseudocode is easier to read and write and allows the programmer to concentrate on the logic of the problem
  - ✓ pseudocode is really structured English
    - statements are written in simple English
    - each instruction is written on a separate line
    - each set of instructions is written from top to bottom, with only one entry and one exit
    - groups of statements may be formed into modules, and that group given a name

1.b Writing Pseudocode

Basic computer operations

- There are six basic computer operations

1. a computer can receive information
2. a computer can put out information
3. a computer can perform arithmetic
4. a computer can assign a value to a variable or memory location
5. a computer can compare two variables and select one of two alternate actions
6. a computer can repeat a group of actions

1.b Writing Pseudocode

Basic computer operations

1. A computer can receive information

- ✓ Get is used when the algorithm must receive input from the keyboard:
  - Get filename
  - Get class number

- ✓ Read is used when the algorithm must receive input from a file:
  - Read course description (from file)
  - Read student names (from file)

1.b Writing Pseudocode

Basic computer operations

2. A computer can put out information

- ✓ Print is used when the output must be sent to the printer:
  - Print 'Program Completed'
- ✓ Write is used when the output must be written to a file:
  - Write student names
- ✓ Display and Prompt are used when the output must be displayed on the screen:
  - Display 'Hello world!'
  - Prompt for class number (generally followed by Get)

1.b Writing Pseudocode

Basic computer operations

3. A computer can perform arithmetic

- ✓ Either actual mathematical symbols or words can be used:
  - Multiply Length by Width to Compute Area
  - Area = Length * Width
- ✓ Words and equivalent symbols used in pseudocode:
  - Add or +
  - Subtract or –
  - Multiply or *
  - Divide or /
  - Modulus or %
  - Parentheses or ( )
- ✓ Compute and Calculate also possible:
  - Compute degrees Celsius
  - C = (F – 32) / 1.8

1.b Writing Pseudocode

Basic computer operations

- Example of pseudocode

Find the mean age of the class

Prompt the user for number_students
Get number_students
Prompt the user for all student ages
Get all student ages
Add all student ages into total_age
Divide total_age by number_students
Display the result

1.b Writing Pseudocode

Basic computer operations

4. A computer can assign a value to a variable or memory location
   - ✓ Initialize or Set are used to give data an initial value:
     - Initialize total_price to 0
   - ✓ = or ← are used to assign a value as a result of processing:
     - total_price ← cost_price + sales_tax
   - ✓ Save or Store are used to keep a variable for later use:
     - Save customer_name in last_customer_name

1.b Writing Pseudocode

Basic computer operations

5. A computer can compare two variables and select one of two alternate actions

- ✓ Examples:
  - If it starts to rain, I'll go inside the building.
  - If the car starts, I'll drive. Otherwise I'll take the bus.
- ✓ Keywords:

```
IF student is female THEN
    Add 1 to female count
ELSE
    Add 1 to male count
ENDIF
```

```
IF age > 0 THEN
    Add 1 to number_students
ELSE
    Display "Impossible"
ENDIF
```

1.b Writing Pseudocode

Basic computer operations

6. A computer can repeat a group of actions

- ✓ Examples:
  - pour water in the saucepan until it is full
  - cook the pasta until it is “al dente”
- ✓ Keywords:

```
WHILE water_level < pan_height THEN
    Add 1 tablespoon to water_volume
    water_level = water_volume / pan_surface
ENDWHILE
```

1.b Writing Pseudocode

Basic computer operations

- Example of pseudocode

Find the mean age of the class

Prompt the user for number_students
Get number_students
Prompt the user for all student ages
Get all student ages

Add all student ages into total_age

Divide total_age by number_students
Display the result

```
set total_age to 0
set count to 0

WHILE count < number_students
        get next student's age
        total_age = total_age + age
        add 1 to count
ENDWHILE
```

1.b Writing Pseudocode

The structure theorem

- Structure theorem
  - ✓ it is possible to write any computer program by using only three basic control structures that are easily represented in pseudocode:
    - sequence
    - selection
    - repetition
- Sequence
  - ✓ straightforward execution of one processing step after another
  - ✓ sequence of pseudocode statements

1.b Writing Pseudocode

The structure theorem

- **Selection**
  - ✓ condition and choice between two actions, depending on whether the condition is true or false
  - ✓ represented by the pseudocode keywords IF, THEN, ELSE, and ENDIF
- **Repetition**
  - ✓ block of statements to be executed repeatedly, as long as a condition is true
  - ✓ represented by the pseudocode keywords WHILE and ENDWHILE

Computer Science I

CS 135

1. Introduction to Programming

a. How to Develop a Program

b. Writing Pseudocode

- ✓ Why pseudocode?
- ✓ Six basic computer operations
- ✓ The structure theorem

c. First Elements of C++

d. Looking Under the Hood