SCSJ 2733
Programming for Engineers

Introduction

“Computers make very fast, very accurate mistakes”

What is Programming

- Programming is writing instructions for a computer to perform.
- But the problem is, a computer takes everything literally and without question.
- It cannot distinguish between what we say and what we might actually intend to say.
- So, humans must be clear and unambiguous when giving instructions (programming) to a computer.

Chicken Curry Recipe

- Ingredients
  - 6 spring onions
  - 3 garlic cloves
  - 2 tbsp vegetable oil
  - half a 400g tin chopped tomatoes
  - 2 tbsp curry powder
  - 1 tsp ground ginger
  - 400g boneless skinless chicken thigh, cut into 2.5cm pieces
  - 100ml Greek-style natural yoghurt, plus extra to serve
  - salt and pepper

Chicken Curry Recipe

- Method
  - Thinly slice the spring onions, reserving a handful of the sliced green parts for garnish. Peel and chop the garlic. Heat the oil in a large saucepan over a medium heat and cook the spring onions and garlic for a few minutes. Add the tomatoes, curry powder and ground ginger and cook for 3-4 minutes. If the pan gets dry add a splash of water and make sure the spices don't burn.
  - Add the chicken and cook for 5 minutes. Make sure all the chicken is coated and is beginning to brown on the sides.
  - Add 250ml water and bring to the boil. Reduce to a medium to low heat and cook for 10-15 minutes, or until the chicken is cooked through with no sign of pink juices in the middle of the pieces.
  - Take the curry off the heat, stir in the yoghurt then season with salt and pepper. Serve the curry with the rice and garnish with a drizzle of yoghurt.

What do we normally program?

- Repetitive & Duplicative process
- Sequence/series/multiple of complex operations
- Large scale computations
- Repeated applications
- Computationally expensive for human to do

Programming vs Natural Language

- **Elements of language –** vocabulary, rules/grammar, structure.
- **Natural languages**
  - can be ambiguous and make small errors, and still expect their intent to be understood
  - human can guess the ‘intended’ meaning

Programming vs Natural Language

- **Programming languages**
  - require a greater degree of precision and completeness
  - have syntactic and semantic rules used to define meaning
  - computers do exactly what they are told to do, and cannot understand the code the programmer "intended" to write
  - are used to facilitate communication about the task of organizing and manipulating information, and to express algorithms precisely

Talking to Computer

- **Machine Code/Language –**
  - The lowest-level programming language understood by a computer’s CPU, consisting entirely of numbers (binary numbers), hardly understood by human.
  - Every CPU model has its own machine code.

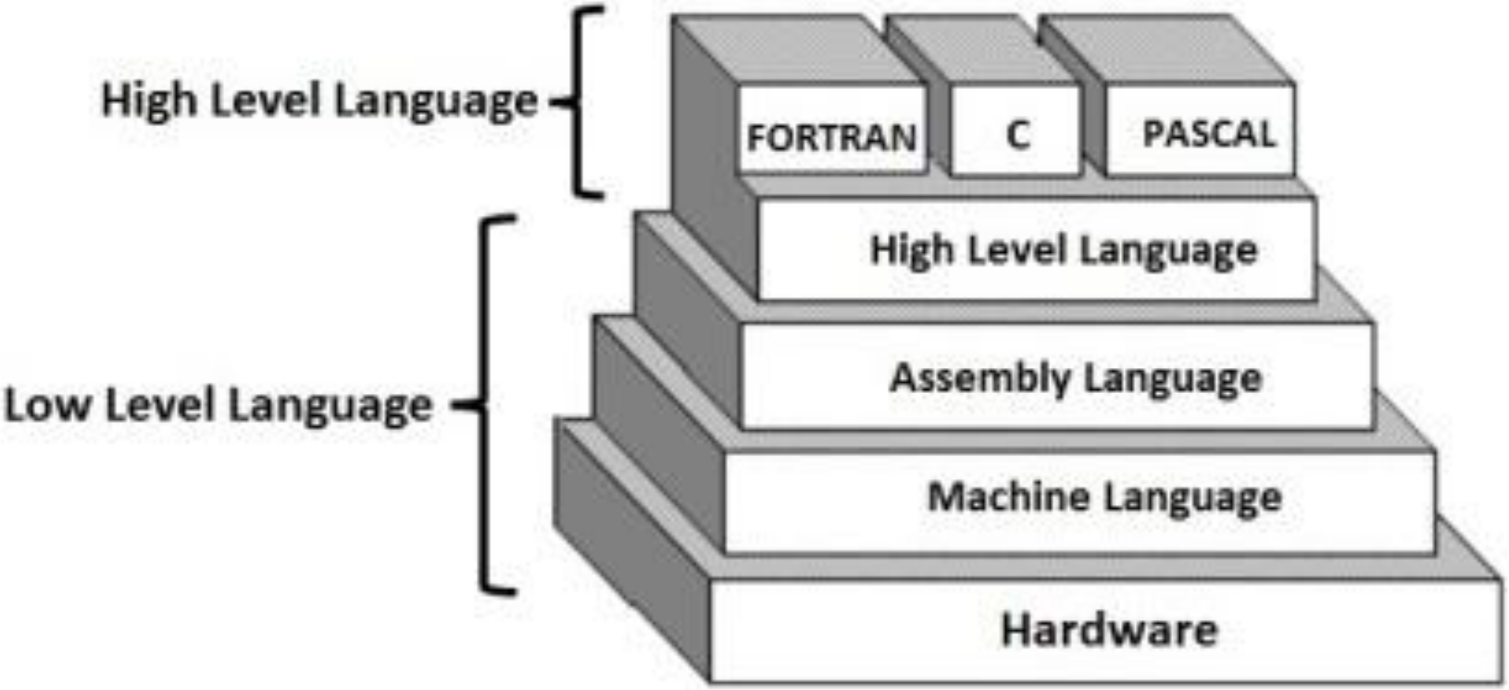
Talking to Computer

- **Assembly Language**
  - An assembly language contains the same instructions as a machine language, but the instructions and variables have names instead of being just numbers

Talking to Computer

- **High-Level Language**
  - A programming language (such as C, FORTRAN, or Pascal) that enables a programmer to write programs that are more or less independent of a particular type of computer. Such languages are considered high-level because they are closer to human languages and further from machine languages.

Talking to Computer



Compiling

- Programs written in a high level language has to be *compiled* (translated) by a *compiler* into machine language (consisting of just binary numbers) before it can be *executed* by the computer.
- Hence, the compiled, ready to run programs are also called *binaries*, or *executables*.

Programming

- Problem steps must be able to be fully & unambiguously described
- Problem types;
  - Can be clearly described
  - Cannot be clearly described (e.g. Beauty)
- Many similarities to solving 'word problems'
  - Translate prob. description into a formal solution
  - Symbol manipulation
- Mix of high level creativity & low level details
- Modularize (for reuse) & Automate (loops)

Steps in Problem Solving (Programming)

- Program Analysis & Specification
- Data Organization & Algorithm Design

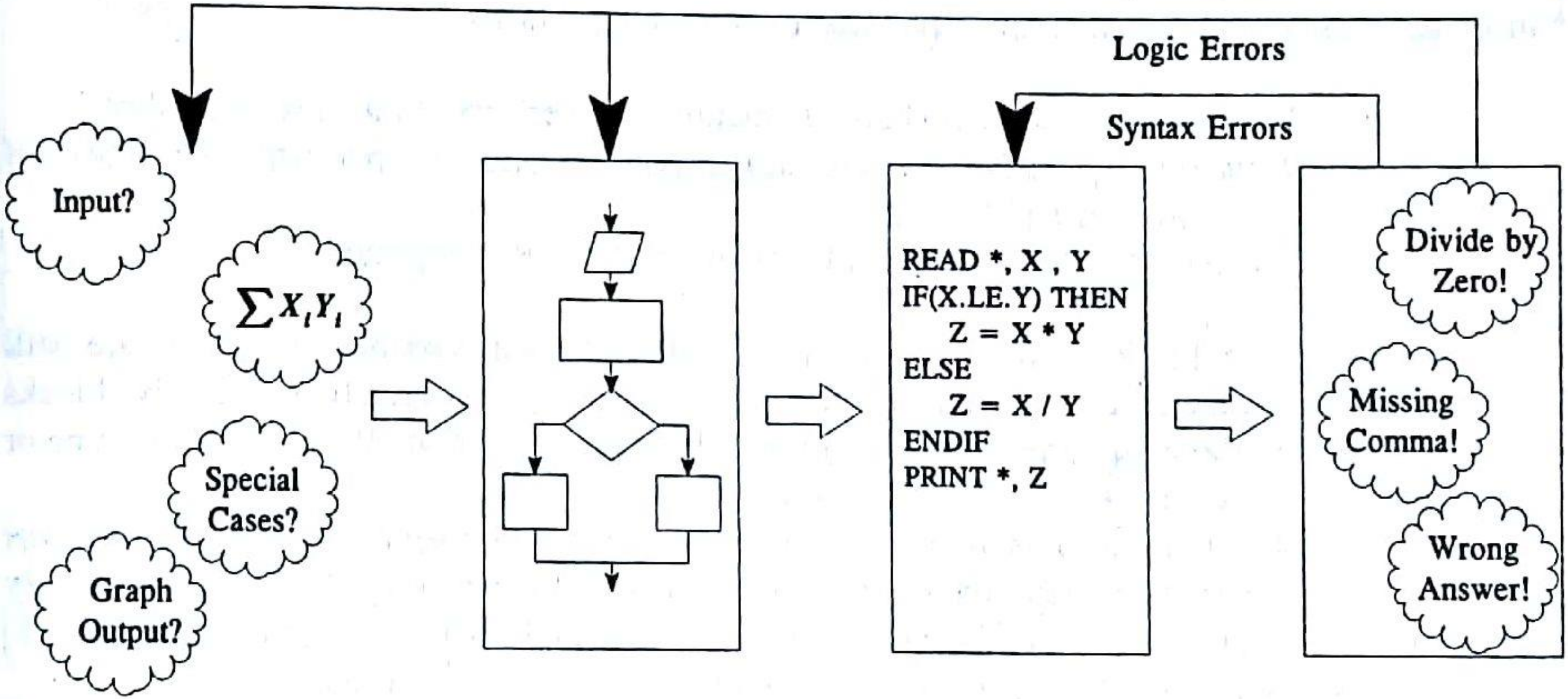
} Logic of Program

- Program Coding
- Execution & Testing

} Coding (Typing)

- Program Maintenance

Stages of Program Development



```
READ *, X , Y
IF(X.LE.Y) THEN
    Z = X * Y
ELSE
    Z = X / Y
ENDIF
PRINT *, Z
```

(a) Program Concept (b) Algorithm Development (c) Program Coding (d) Debugging

Fig. 1-1 Stages of program development

Prog. Analysis & Specification

- Specs must include
  - Input – specs. Of problem input
  - Output – description of problem's output
- May be complex, imprecise, vague – but must be clarified at this stage

Data Org. & Algorithm Design

- Problem's Data – Appropriately Structure & Organize (to store data)
- Design Algorithm – how to process data
- Algorithm – Precise sequence of simple steps to arrive at solution
    - May be written in a mix of pseudocode & flowchart
        - Pseudocode – mixture of natural language, symbols, programming language of choice

Structured Algorithm / Programs

- Use 3 basic methods of control
  - **Sequential** : steps done in sequential manner, each step executed once
  - **Selection** : One of a number of alternatives is selected and executed (if / condition)
  - **Repetition** : One or more steps are performed repeatedly (loops)
- For more complex problems – divide & conquer
  - Break into smaller, manageable subprograms (modules)

Program Coding

- Coding – Implementing data objects & algorithms in a chosen programming language
- Syntax – must follow grammatical rules of that language
  - Variables
  - Types : real, integer, etc.
  - Operations : +, -, x, /, **
  - Assignment : =
  - Input/output : read, print, write
  - Comments

Programming Style (Coding)

- Correct, Readable, Understandable. To achieve these;
  - Programs must be well-structured
    - Top-down approach (divide & conquer)
    - Simplicity & clarity
  - Each program unit must be documented (comments)
    - Opening comments (purpose, data, author, date, ref.)
    - Key program segments
    - Meaningful identifier (variable names)

Programming Style (ctd.)

- Formatted to increase readability
  - Spaces
  - Blank lines between sections of program
  - Alignment & indentations
  - Choice of fonts
    - Fixed-width font (programmer’s font)
    - Ex.: Courier, Consolas, Inconsolata

Execution & Testing

- Program must be correct (produce correct results)
  - Validation – program meets project specs. (answering the question)
  - Verification – results are correct & complete (answering it correctly)

Errors (Bugs)

- Syntax error
  - Compile time error (missing comma, brackets etc)
  - Runtime error (divide by zero, infinite loop etc)
- Logic error
  - From flaw in algorithm

Program Testing

- Test with simple cases which results are known in advance (reality check)

Program Maintenance

- Maintain Flowchart
- Maintain Source Code
- Make code readable