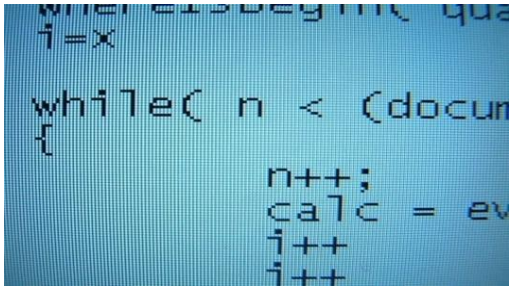
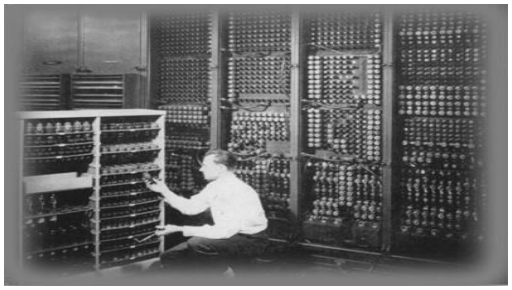
University of Reading





Taster Session for the Year 1 Module: Programming

Introduction to Algorithms

Definition: Algorithm

University of Reading

*A **procedure** for **solving** a mathematical problem **in a finite number of steps** that frequently involves the repetition of an operation*

Broadly: a **step-by-step procedure for solving a problem**

or accomplishing some end

[Merriam-Webster]

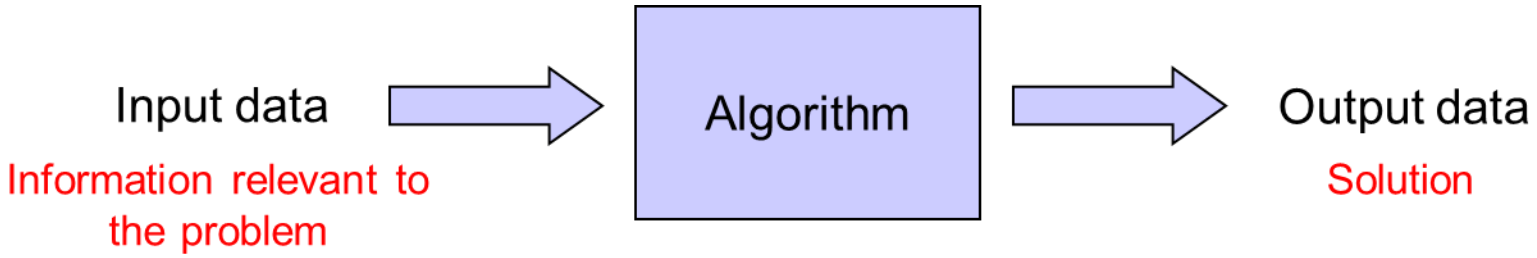
Algorithms are the threads that tie together most of the subfields of computer science.

[Donald Knuth]

Definition: Algorithm: More Precise

University of Reading

An algorithm is a sequence **of unambiguous executable steps, defining a terminating process for solving a problem**



- Input: inputs, taken from a specified set of objects
- Output: one or more outputs with a specified relation to the inputs

- Properties:
  - **Definiteness**: Actions must be precisely and unambiguously specified
  - **Effectiveness**: All actions must be sufficiently basic that they can be done exactly and in a finite length
  - **Finiteness**: must have an end, thus, indeed produce some kind of output

Question Time

University of Reading

- Think a moment for yourself
- Can you give an example for an algorithm?
  - Maybe you executed an algorithm?
  - If so, what "algorithm"?
- Time: 1 min

Example Algorithms

University of Reading

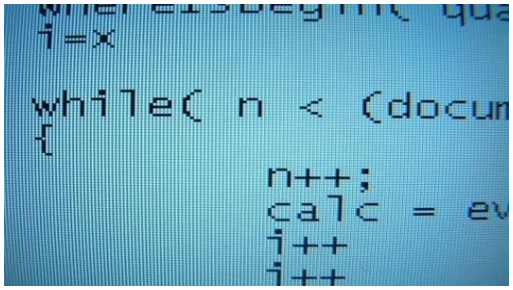
- I'm sure you applied algorithms! They are everywhere!
- Your activities every day
  - Following recipes: How to make a cake
  - Showering, coffee making, driving a car
- Embedded systems
  - The movement of a lift (when pushing a button from outside)
  - Biometric identification on your smartphone
- Maths
  - Given two numbers, identify the greater one
  - Compute the circumference of a circle or perimeter of a rectangle

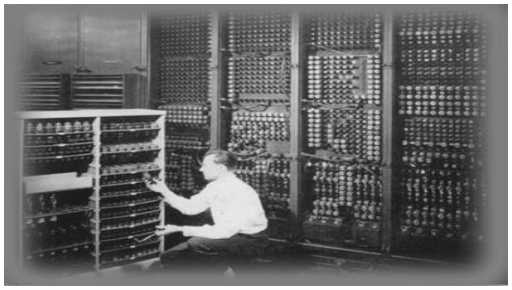
Additional Properties of Algorithms

University of Reading

- **Correctness**
  - Must return the desired output for all legal instances of the problem
  - It is always assumed that an algorithm is correct!
- **Machine-Independent:** Stays the same independent of
  - Which kind of "hardware" will execute it (or if manually executed)
  - Which description or programming language it will be written in
- **Efficient**
  - Can be measured in terms of resources, i.e., in computer science
    - Running time (**time complexity**)
    - Memory requirement (**space complexity**)

University of Reading





The Structured Programming Theorem

Structured Programming Theorem

(Böhm-Jacopini Theorem; in simplified form)

University of Reading

All algorithms of computable functions can be specified by composing elementary **tasks (actions)** into more complex tasks via the constructs:

- **Sequence**: Executing actions consecutively (one, then the next, …)
- **Selection**: Executing one of two actions according to the value of an expression
- **Iteration**: Repeatedly executing actions as long as an expression is true

[Wikipedia]

Sequence

University of Reading

- Sequence simply performs one step after another
- Each step is executed in a specific **sequence**
  - do this
  - then do this
  - then do this
- Even better, we can explicitly use numbering
  - Example: (problem: thirsty)
    1. Open the fridge
    2. Take the milk
    3. Drink the milk
- It is the designer of the algorithm who decides about the order
  - In the "milk" example, any different order wouldn't work

Selection

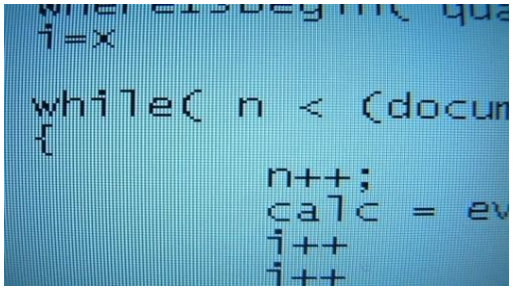
University of Reading

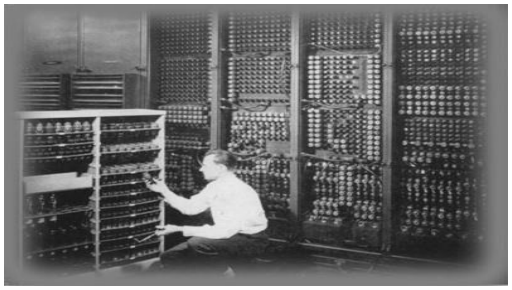
- Selection is the decision-making construct
- It is used to make **yes/no** or **true/false** decisions logically
- Can be considered as a condition:
  - if something is true, then take **this action**
  - otherwise, take **that action**
- Example
  - if it rains, then take a raincoat and an umbrella, else take neither
- Selection is called **conditional statement** in programming languages

Iteration

University of Reading

- Iteration repeats a sequence of actions based on a condition
  - Iteration comes from the word reiterate, which means to repeat
  - Iteration is a **looping** construct
- Two styles of iteration:
  - Explicit number of iterations
    - Repeat something n times
    - Example: Repeat this X times, where X is the number of people in this room
  - Implicit number of iterations
    - While something is **true**, keep **doing this,** otherwise stop
    - Example: While it is sunny keep playing in the garden





Describing an Algorithm

Describing an Algorithm

University of Reading

- Descriptions are important to share/understand algorithms
  - The structured programming theorem simplifies how to describe algorithms
    - Just need to define a representation of Sequence, Selection, and Iteration
- There are various ways of expressing an algorithm:
  - Textual:
    - Natural language, i.e., in English text
    - Pseudocode, i.e., programming code alike description
    - Programming language
  - Graphical
- The quality of representations differ for different notations
  - All must share algorithm properties: definiteness, effectiveness, finiteness
  - **Understandability and standards are important**

Natural Language

University of Reading

Recipes: More or less standardised description

Hummus and Tomato Pasta
Serves 2
This is a very quick 20-minute after work dish.

Ingredients:
Olive oil
1 teaspoon of whole cumin seeds
1 large chopped onion
400g chopped plum tomatoes
200g hummus
150g pasta

Steps:
1. Add the pasta to a large pan of boiling water. Simmer for 10 minutes.
2. Fry the cumin in the olive oil for two minutes. Add the onions and fry gently.
3. Stir in the tomatoes and the hummus and leave to simmer until done.
4. Serve the pasta and add the sauce.

Based upon http://www.cs4fn.org/programming/recipeprogramming.php

Group Work: Discussion

University of Reading

- Question: Is this a "good" description for an algorithm?
  - **Does it meet the requirements**
    - **Definiteness:** is each step precisely defined?
    - **Effectiveness:** is each step basic enough and can be executed in practice?
    - **Finiteness:** does it terminate?
  - **How "understandable" is the algorithm for You? Can you execute it?**
- Time: 3 min
- Sharing: 1 min

Hummus and Tomato Pasta
Serves 2
This is a very quick 20-minute after work dish.

Ingredients:
Olive oil
1 teaspoon of whole cumin seeds
1 large chopped onion
400g chopped plum tomatoes
200g hummus
150g pasta

Steps:
1. Add the pasta to a large pan of boiling water. Simmer for 10 minutes.
2. Fry the cumin in the olive oil for two minutes. Add the onions and fry gently.
3. Stir in the tomatoes and the hummus and leave to simmer until done.
4. Serve the pasta and add the sauce.

Discussion of the algorithm

University of Reading

Hummus and Tomato Pasta
Serves 2 **[Useful to know, kind of the output]**
This is a very quick 20-minute after work dish. **[Describes the "performance"]**
Ingredients: [Useful to know, kind of the input]
Olive oil
1 teaspoon of whole cumin seeds
1 large chopped onion
400g chopped plum tomatoes
200g hummus
150g pasta

Steps:
1. Add the pasta to a large pan [ok, large to cover all pasta] of boiling water. Simmer for 10 minutes.
2. Fry the cumin in the olive oil [in another pan?] for two minutes. Add the onions and fry gently.
3. Stir in the tomatoes and the hummus [to which pan? The sauce?] and leave to simmer until done [How do I know when it is done?].
4. Serve the pasta and add the sauce [The sauce is in the second pan?].

Definitness? Understandability?

University of Reading

- This video shows "literal" execution of instructions for making a Peanut-Butter-Jelly sandwich
- https://youtu.be/cDA3_5982h8?t=19
- Thus, it is not trivial to write down algorithms!

Pseudocode

University of Reading

- Pseudocode is a semi-formal representation of an algorithm
  - Pseudo means "pretend" or "false"
- Pseudocode pretends to be computer code
  - It uses **constructs** of programming languages
- Pseudocode syntax is not well standardised
  - There are various forms of pseudocode
    - Some are closer to a programming language
    - Some are closer to natural language

Pseudocode: How to Start a Car

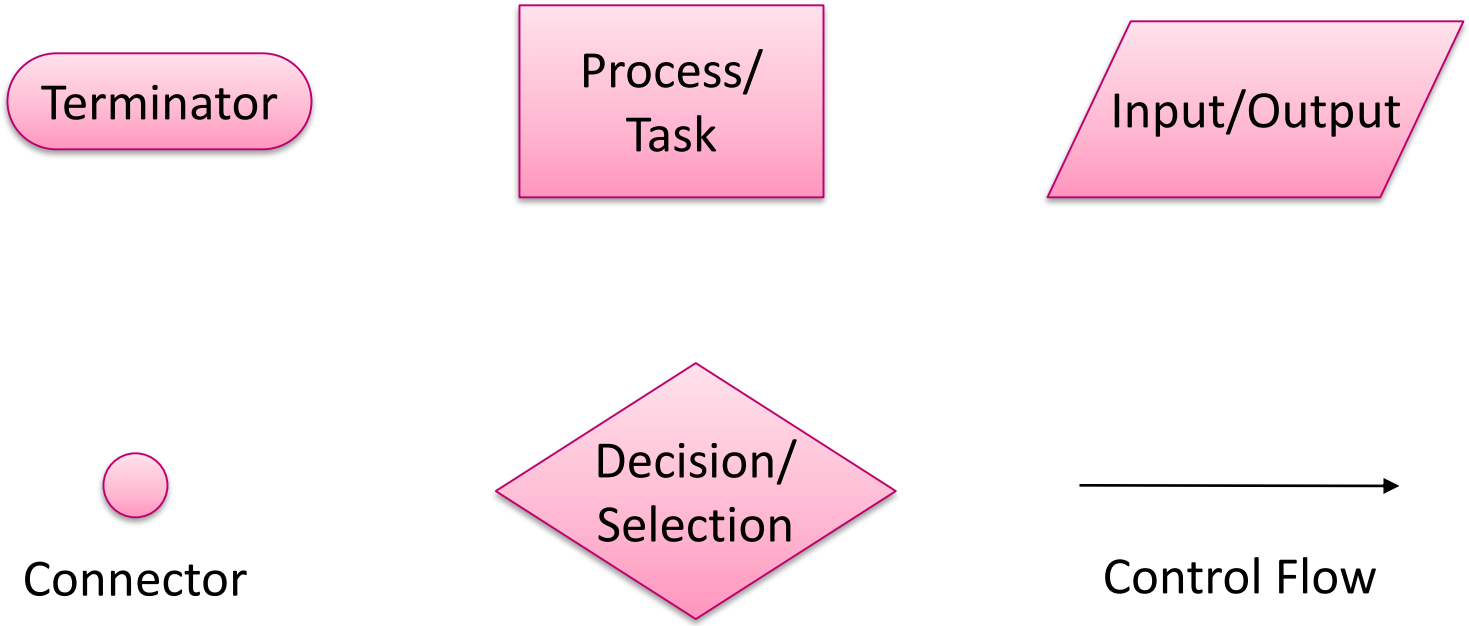
```
Insert Key in Ignition
Turn Key to Start Position

while Engine hasn't started do
    Hold Key in Start Position
    Wait 3 seconds
    if Engine hasn't started then
        Turn Key to Off Position
        Wait 10 Seconds
        Turn Key to On Position
    end
end
```

Flowcharts

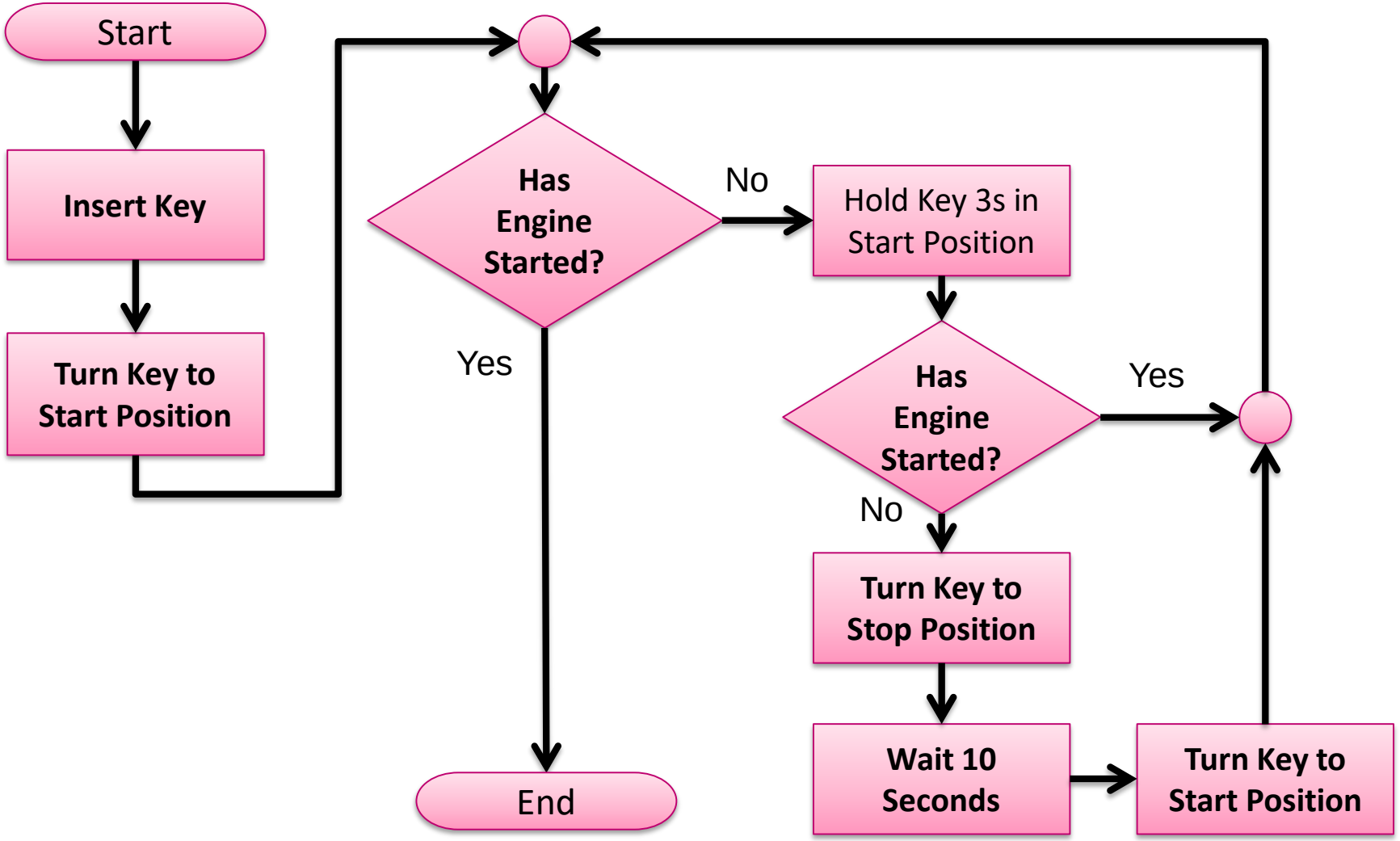
University of Reading

- A Flowchart is a visual representation of an algorithm
  - Alternative to pseudocode, serves the purpose of communication
- Flowcharts are well-defined and standardised
  - Use easy-to-understand symbols to represent steps
  - Visual elements represent the constructs of programming languages
- Symbols:



Flowchart Examples: Start a Car

University of Reading



Summary

University of Reading

- Every algorithm can be coded using:
  - Sequence, Selection, Iteration
- Algorithm:
  - Sequence of unambiguous executable steps, defining a process for solving a problem
  - Mandatory requirements: **Definiteness, Effectiveness, Finiteness**
  - Properties: Correctness, Machine-Independent, Performance
- Describing Algorithms:
  - Natural language, Pseudocode, Flowchart
- Outlook:
  - A *programming language* is a description that is executable