Software Development 1
Introduction

TU Dublin –Tallaght Campus

Eileen Costelloe

Welcome to Software Development 1

Useful Information

Eileen Costelloe	**eileen.costelloe@TUDublin.ie**
Moodle Enrolment ID	SDev1
Language:	Python (**Version 3.6 or higher)**
IDE - **I**nteractive **D**evelopment **E**nvironment	PyCharm **(Version: PyCharm-community.2019.2 or higher)**
Course Book	Python Programming, for the absolute beginner, Third Edition, Michael Dawson

Course Description

This module introduces programming concepts and structures using a problem based learning approach. The fundamentals of a programming language are covered to facilitate the student analysing a problem and then designing and implementing a suitable solution. The development life-cycle from analysis, design, implementation, debugging are covered to ensure the student understands and is able to demonstrate the process

Learning Outcomes:

LO1 Write programs using variables, conditional statements, loops & lists

LO2 Apply problem solving techniques to programming problems

LO3 Use logical and syntactical debugging techniques, management of projects

LO4 Apply good programming practice and style

Note : The full syllabus is on the Moodle course page

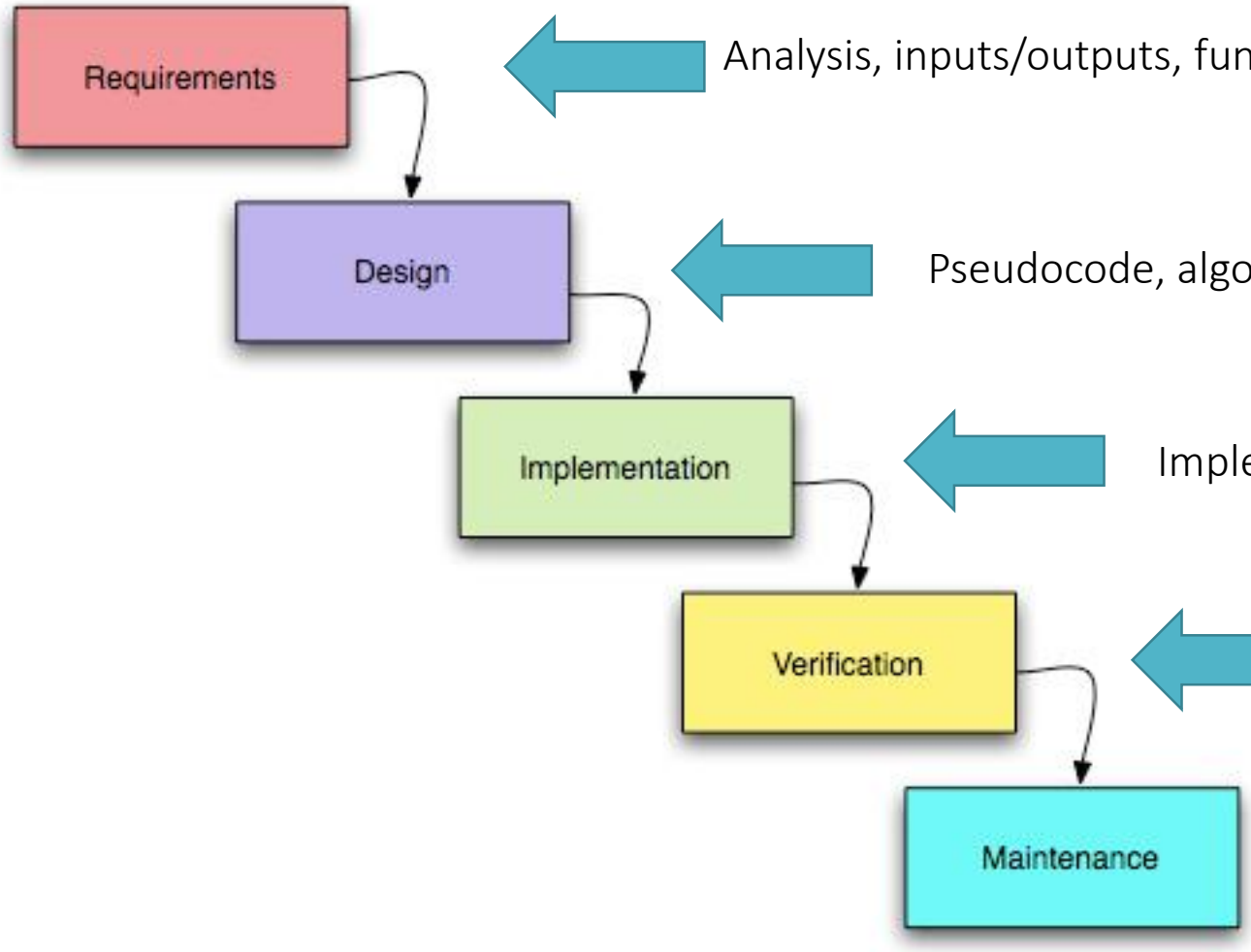
Problem Solving & Algorithms

- Problem solving forms part of thinking.
- It is considered the most complex of all intellectual functions, problem solving has been defined as a higher-order cognitive process.
- Problem Solving is fundamental to Software Development.
- The role of the Software Developer in industry is to write programs that address the needs of the employer, typically involving solving of business problems.

Introduction – Problem Solving

- In Software Development there are processes to aid developers in solving problems and developing systems.
- The Waterfall/Structured Development Method
- Agile Development (Usually Scrum)

Waterfall method



Analysis, inputs/outputs, functionality, users

Pseudocode, algorithms, sequence of instructions, calculations

Implement in a programming language

Test and debug

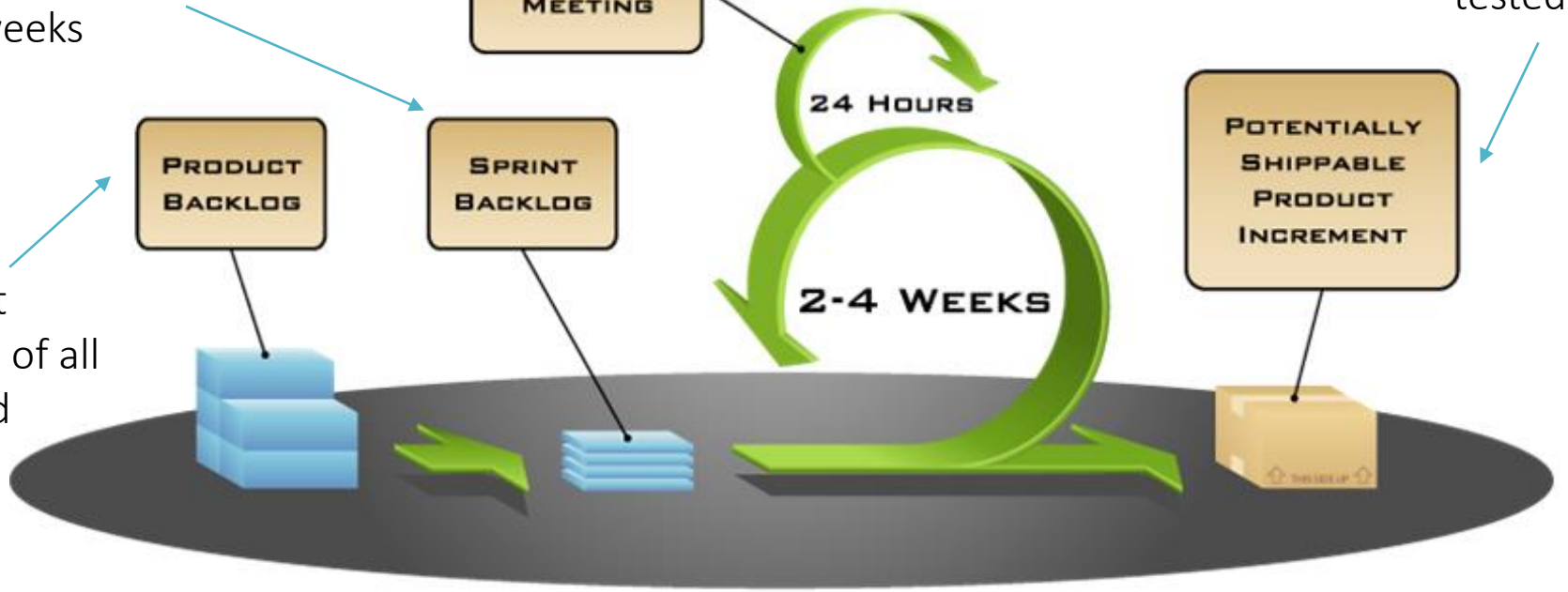
Install ,maintain and update system

Agile (Scrum) method

What's done, to do, issues

List of tasks to be completed in a sprint. A sprint is 1 development cycle, e.g. 2-4 weeks

Functions designed, coded, tested and integrated



Prioritised feature list contains descriptions of all functionality required

Benefits of the Agile approach are the quick feedback and the ability to adapt to changing requirements

Introduction – Pseudocode(part of Analysis & Design phase)

- Pseudocode is a type of structured English for describing algorithms- an algorithm is a set of instructions, a step by step procedure to solve a problem.

- Pseudocode should be generic i.e. it can be applied to any programming language. Therefore no programming language specifics should appear where possible.

- Pseudocode cannot be compiled or executed, and there are no real formatting or syntax rules, although most organisations will have their own conventions for writing it. It is simply one step – an important one - in producing the final code.

Pseudocode Example

Write the pseudocode to get two numbers from the user, add them together and output the result to the screen

Pseudocode:

Get number 1 from the user
Get number 2 from the user
Add the two numbers together and store the result
Display the result

Approach to follow:

1. *Get any required inputs from the user*
2. *Process / perform any required calculations*
3. *Output your results*

Input – Process – Output - IPO

Introduction - Algorithms

What is an Algorithm?

An algorithm is a sequence of instructions to solve a particular problem.

- Lets try one:

✍ *Write an algorithm to make a cup of tea.*

✍ *Assume you have electricity and water available to use.*

Introduction - Algorithms

- What does your algorithm look like?
- This process takes time to master!
- Let’s see a sample solution:

Introduction – Pseudocode - Tea

1. Begin
    2. **Get kettle, water, cup, milk, tea bag, spoon, bin**
    3. Put water in kettle
    4. Place tea bag in cup
    5. Switch on kettle
    6. Wait until water boiled
    7. Pour water from kettle into cup
    8. Wait 10 seconds
    9. Remove tea bag from cup with spoon
    10. Put tea bag in bin
    11. Pour milk into cup
    12. Stir milk into tea with spoon
    13. Remove spoon from cup
    14. **Tea is ready**
15. End

Pseudocode - Template

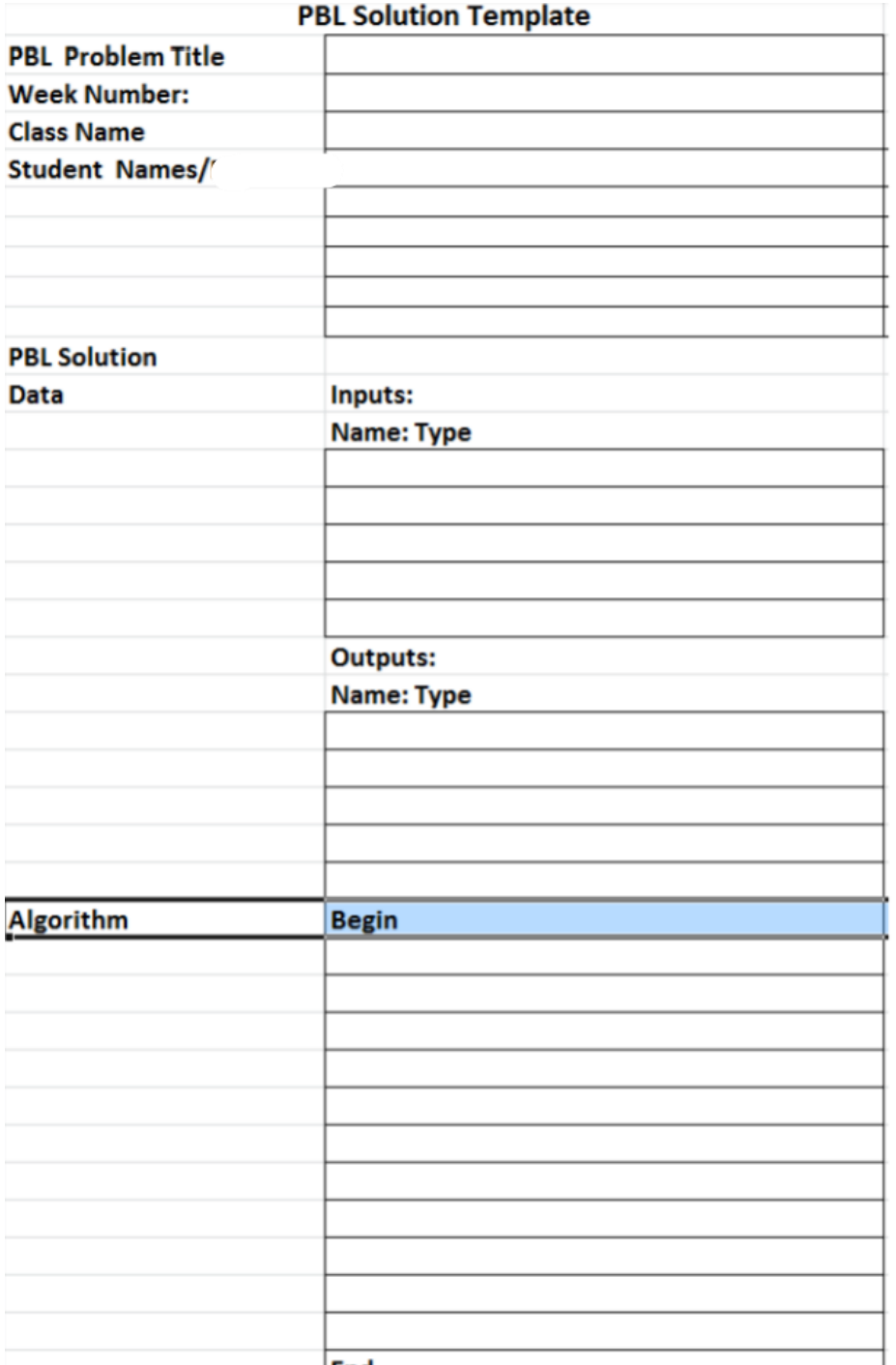
- The template helps formalize the Algorithm Creation Process.
- Useful as programs increase in size.
- Lets step through the headings:

Template:

- **Inputs** — What inputs do you need?
- **Outputs** — What outputs will you generate?
- **Steps** — What is your algorithm, i.e. steps?

Template

- The template helps formalize the Algorithm Creation Process.
- Useful as programs increase in size.
- For PBL (Problem Based Learning) Labs and hand ups.
- Although at the start may take some time, this process will really help speed up the development of larger programs and significantly reduce the errors or bugs.



PBL Solution Template	
PBL Problem Title	
Week Number:	
Class Name	
Student Names/	
PBL Solution	
Data	Inputs:
	Name: Type
	Outputs:
	Name: Type
Algorithm	Begin

Pseudocode / Algorithm Example

Problem: Get 2 numbers from the user, add them together and display the result

Inputs:	**Name**	**Type(of data being stored)**
	number1:	real (any number, whole or fractional)
	number2:	real
Ouputs:	sum_of_numbers :	real

1.Begin

2. **Input** number 1 (assume the value goes into the container number1)

3. **Input** number 2 (assume the value goes into the container number2)

4. sum_of_numbers = number1 + number2 (assume calculation result goes into container called sum_of_numbers)

5. **Output** “Sum of the 2 numbers is“sum_of_numbers (outputs text in quotes and the value stored in sum_of_numbers)

6.End

Algorithm 2

Lets try another example:

✍ *Write an algorithm to get the length and width of a room from a user, calculate the area of the room, and display the area of the room to the user .*

Pseudocode / Algorithm 2

Inputs:	Name	Type(of data being stored)
	room_width:	real (any number, whole or fractional)
	room_length:	real
Ouputs:	area_of_room :	real

1.Begin

2. **Input** width of room (assume the value goes into the container room_width)

3. **Input** length of room (assume the value goes into the container room_length)

4. area_of_room = room_width x room_length (assume calculation result goes into container called area_of_room)

5. **Output** "Area of room is " area_of_room (outputs text in quotes and the value stored in area_of_room)

6.End

A Programing Language

- Formal language designed to express computations
- Syntax – i.e. grammar that governs the structure of statements
- Operations such as Input and Output, Selection(ifs), Repetition(loops), Mathematical operations
- Unambiguous Instructions – a statement just has 1 meaning

- A general purpose language - often described as a Scripting Language
- Leader in Data Analytics, Machine Learning and AI
- Modern, Object Oriented Concepts
- Fast, Light ,Readable, Easy to use language to program – this increases programmer productivity
- Large Libraries included – supports an array of programming tasks, e.g. data analysis, visualisations, gaming
- Portable – most Python programs run unchanged on all major computer platforms, Windows, Linux etc
- Free – open source
- Used by – Google(in its web search systems), YouTube(video sharing service), Dropbox, Intel, Cisco, HP, IBM (hardware testing), NASA(scientific programming tasks) etc.

Python Interpreter- a type of program that executes other programs