University of Strathclyde Glasgow

Programming & variables

Computing & Information Sciences

W. H. Bell

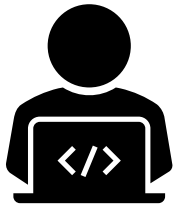
University of Strathclyde Glasgow

Welcome to module

Starting point

University of Strathclyde Glasgow

- Assume no prior programming experience.
  - Start from basics.
  - Introduce ideas quickly.
- Cohort has range of ability and experience.
  - Contribute to class discussions.
  - Support each other, except during assessments.
  - Go beyond class material.



Learning style

Solve by practice.
Avoid documents.
Avoid study.
Avoid discussion.

Try ideas and learn from mistakes.
Refer to documentation as needed.
Discuss with others.

Read documents.
Avoid programming.
Fear failure.

Interactions

University of Strathclyde Glasgow

- Raise questions quickly.
  - If something is confusing, it might confuse others too.
  - Test knowledge by writing programs.
- Use Mattermost and Forums effectively.

Objectives

University of Strathclyde Glasgow

- Learn to write a computer program.
  - Logic.
  - Structure.
  - Debugging.
- Learn basics of Python programming language.
- Write procedural and objected-orientated programs.

Resources

- Lectures
  - Concepts and ideas.
- Worked examples.
  - Trying out the ideas.
- Lab exercises.
  - Understanding the concepts.
- Supporting reading.
  - Solidifying and broadening learning.

Assessment

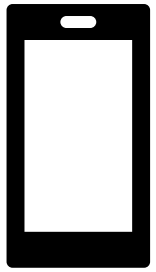
Individual assessments. Must not collaborate with other students.

- Class test – 50%.
  - Closed-book multiple choice quiz.
- Programming exam – 50%.
  - Exam conditions, with access to MyPlace.
  - Visual Studio Code and Python in CIS Linux lab.

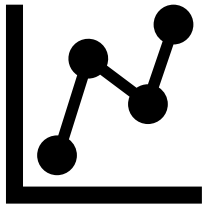
Motivation and context

Enabling solutions

New software

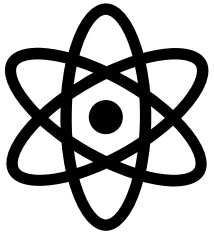


Mobile applications



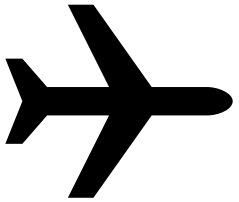
Data analysis

New ideas

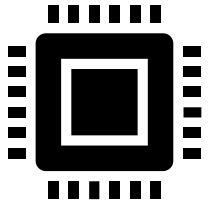


Energy

Automotive

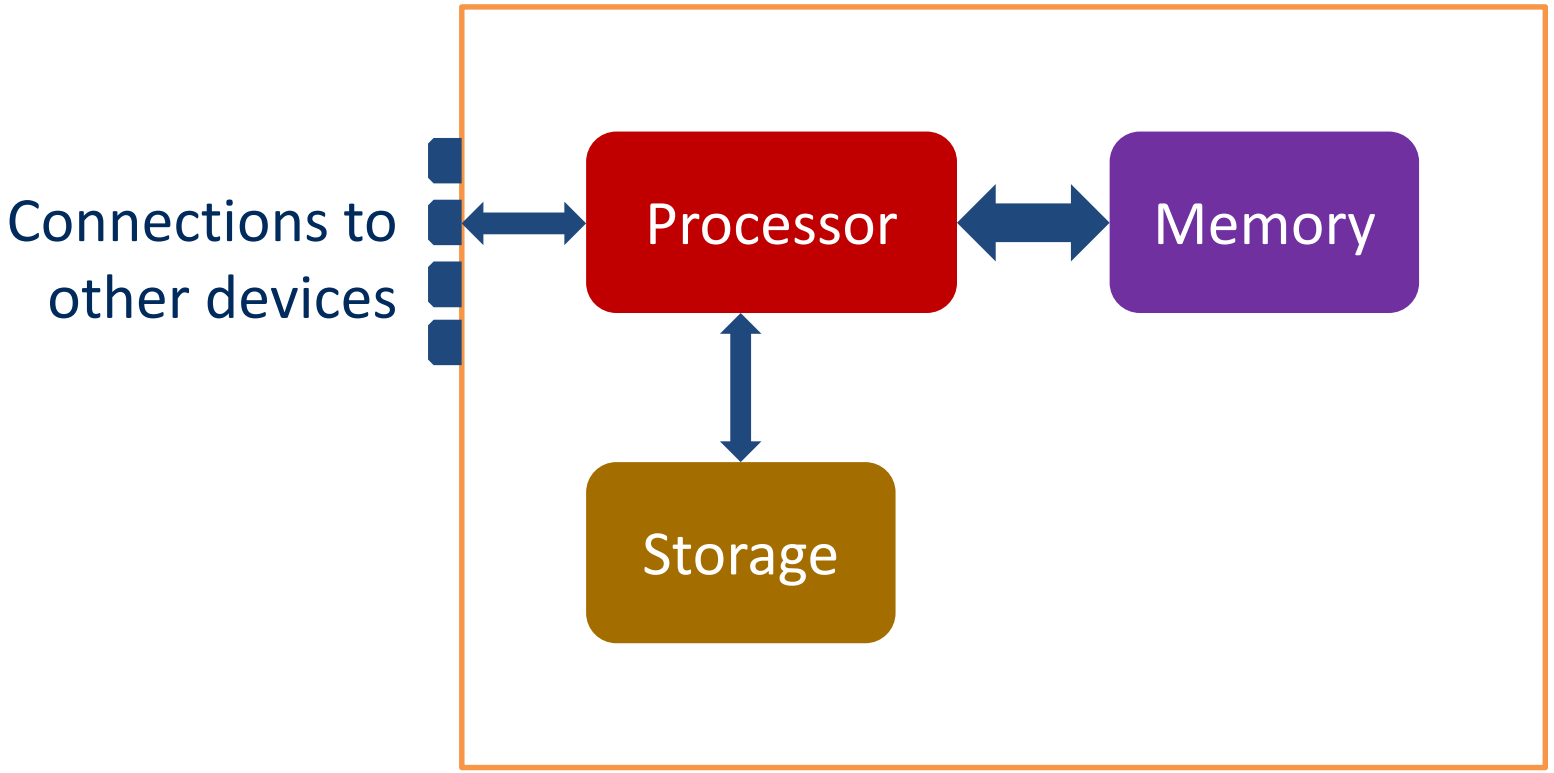


Aviation



Embedded solutions

Computer architecture



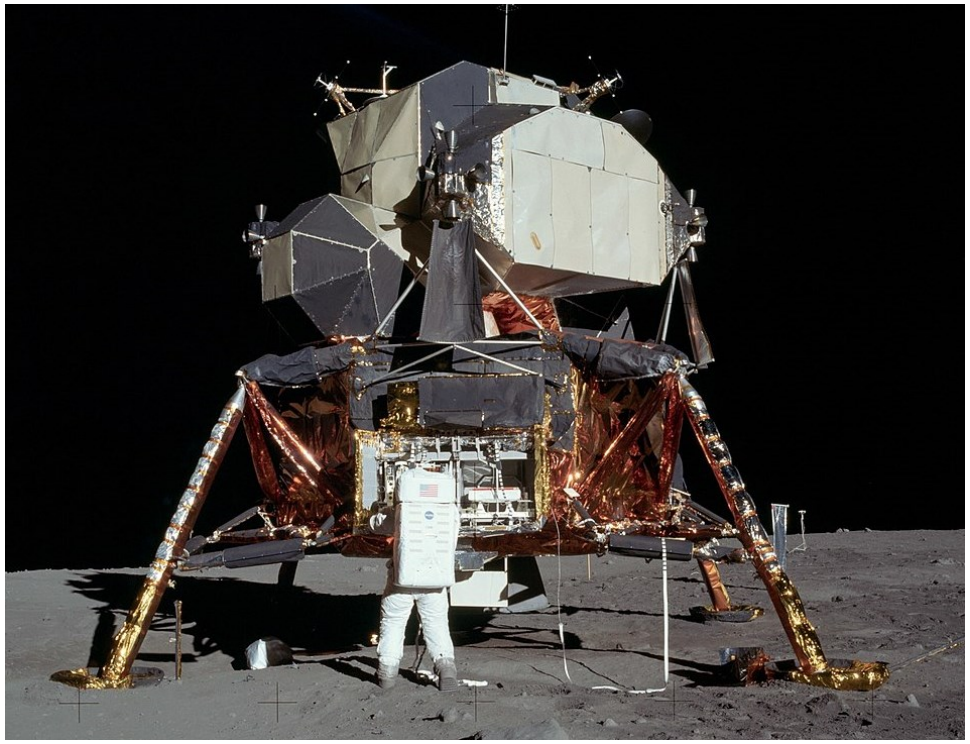
Programming languages

Software engineering

Apollo guidance computer

Free flow ideas → Structured thought process

DSKY user interface



Apollo 11: Luna module

Thought process

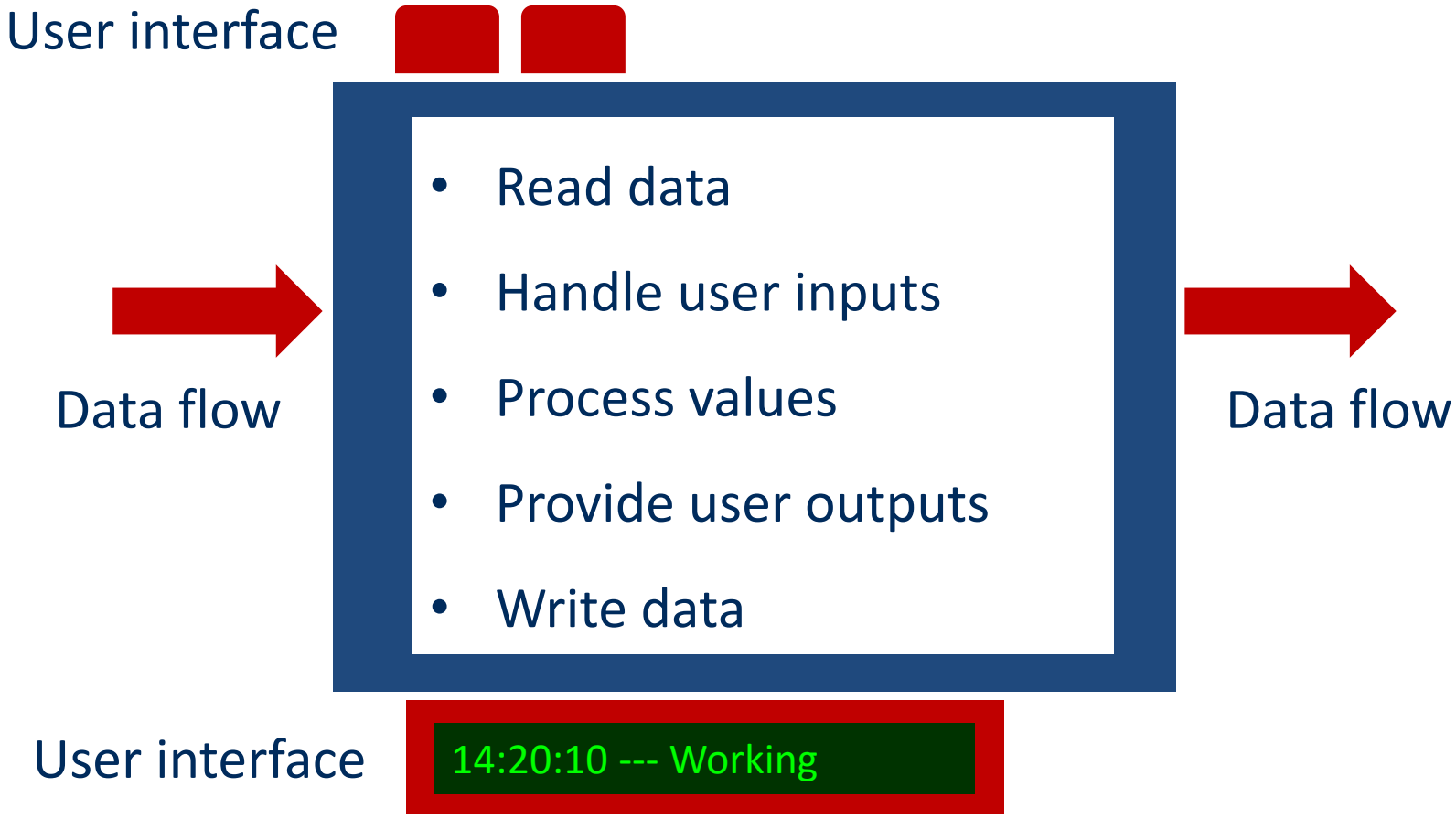
- Define the problem – user requirements.
- Define the user interface.
- Define the high-level design.
- Define the components of the program.
- Implement the program.
- Test the program – check requirements are fulfilled.

First thoughts

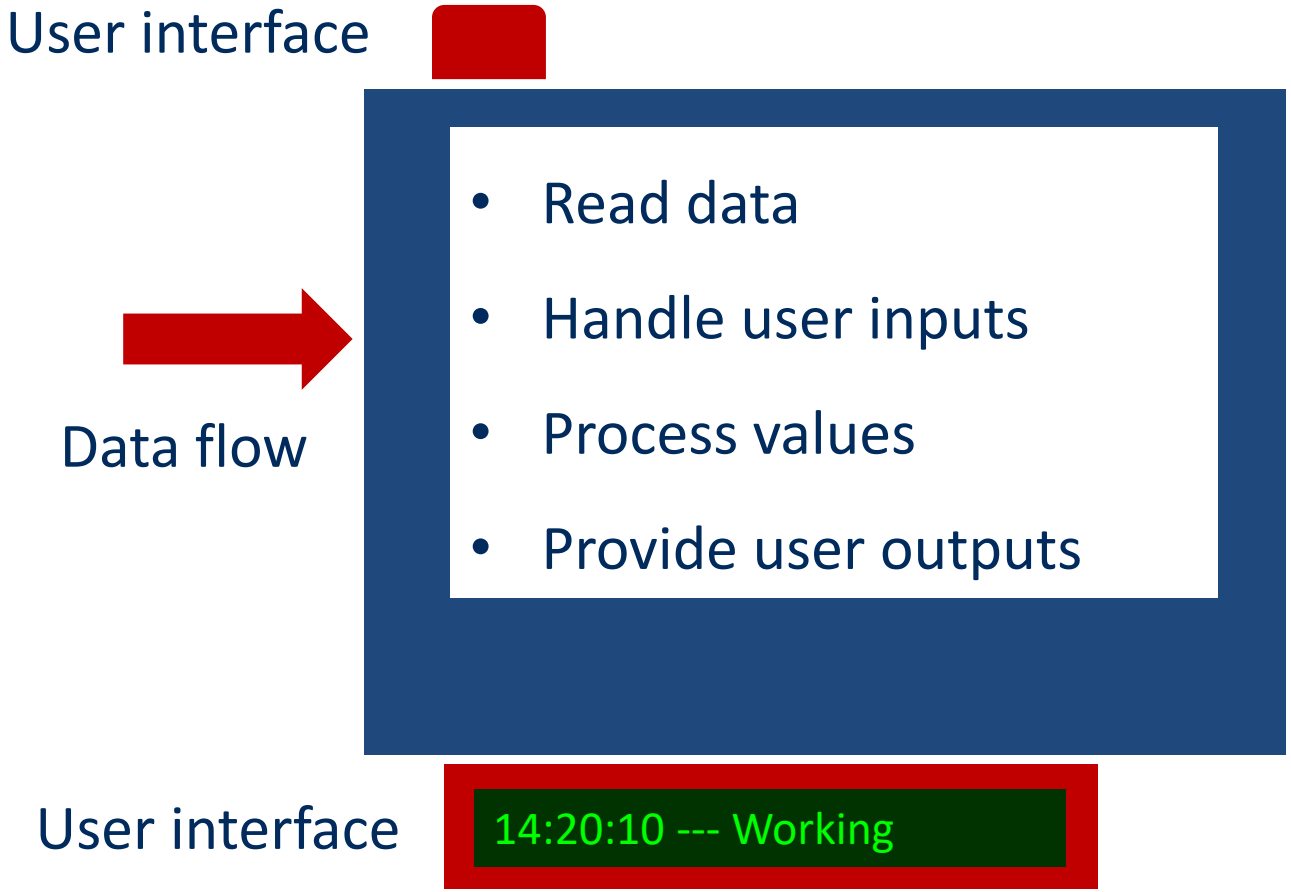
Solve my problem!

Capturing user requirements

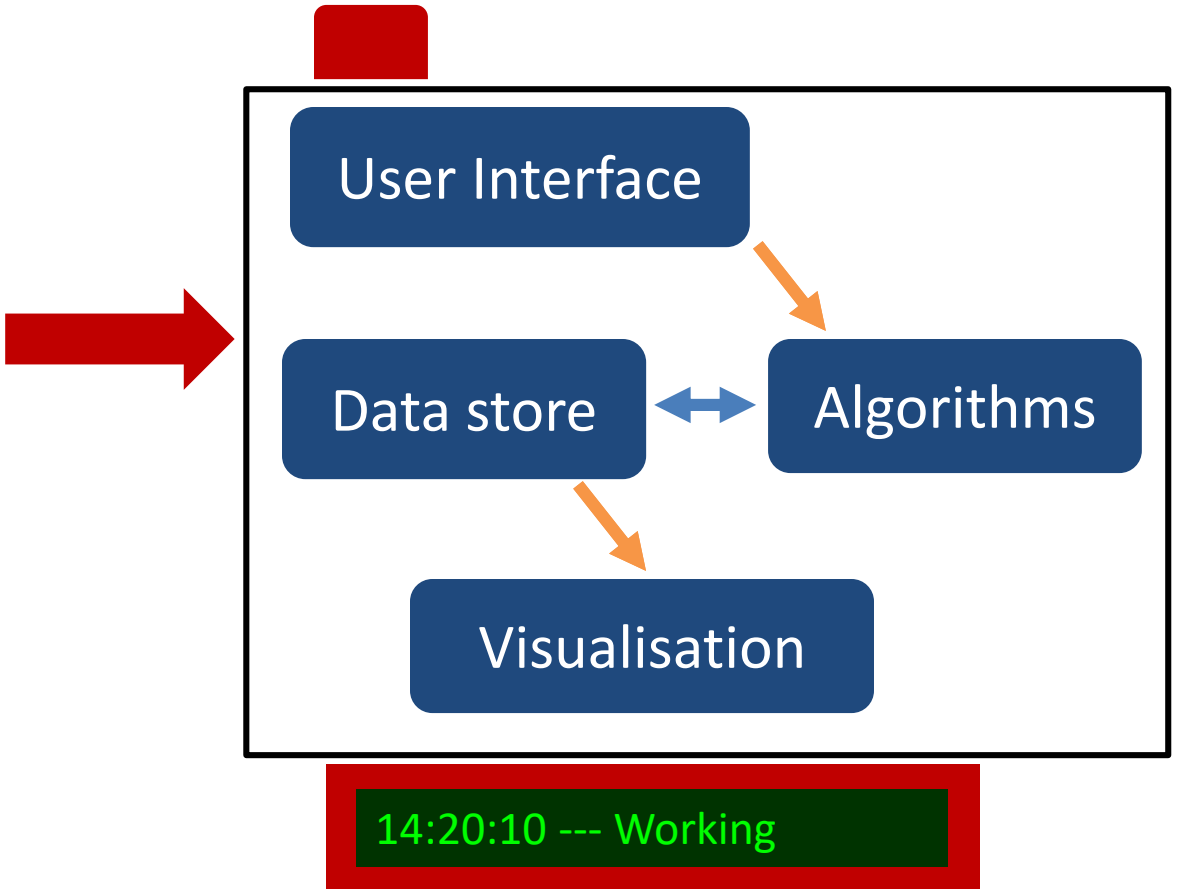
Capturing user requirements



Minimum viable product



High-level design



Ordering thoughts

University of Strathclyde Glasgow

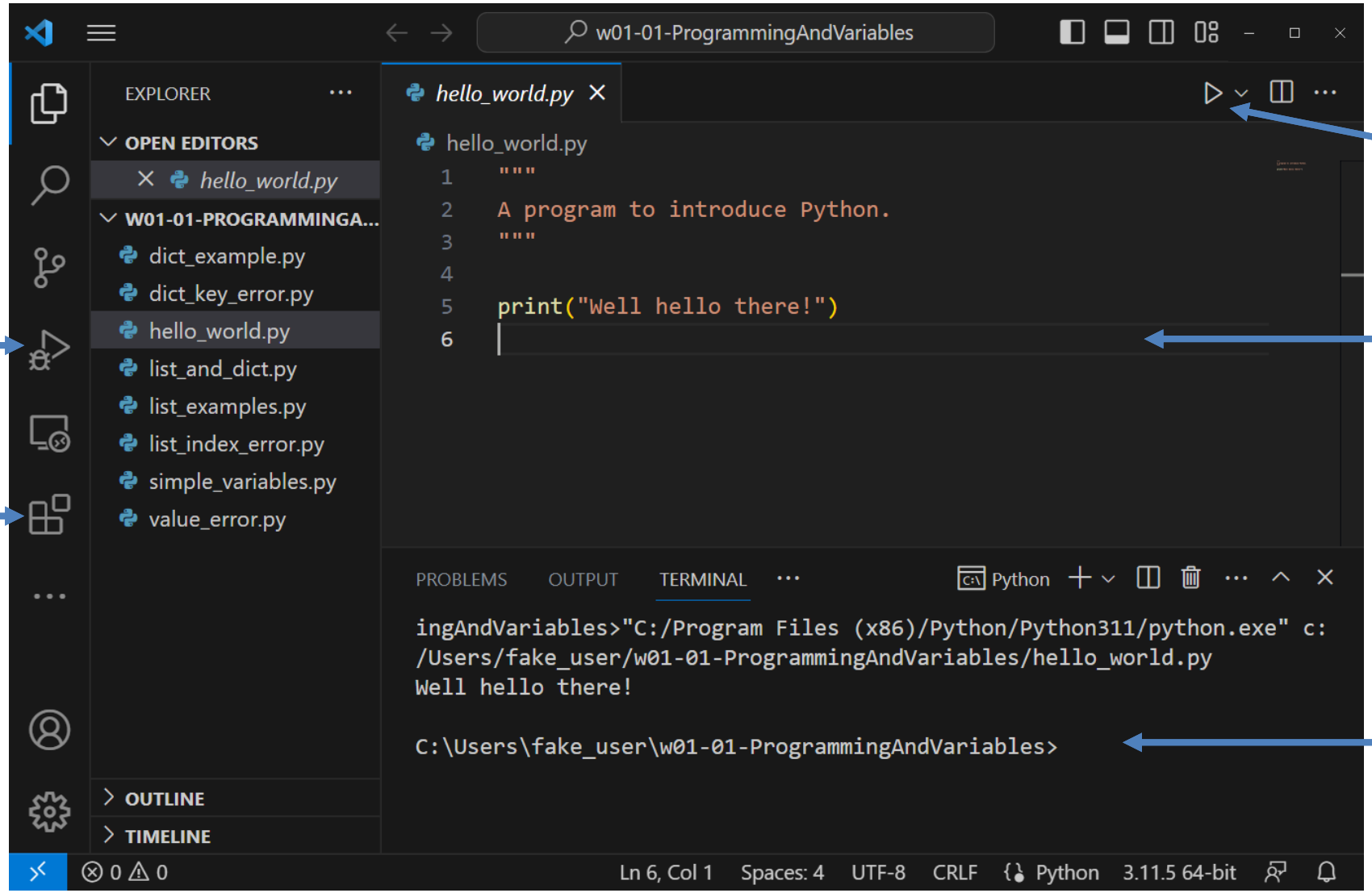
- Flowcharts and process flow.
  - Useful for algorithms or conveying high-level requirements.
- Pseudocode.
  - Create structure in language that is not final code.
  - Fast and can re-use for source code comments.
- Record thoughts.
  - Break program down into small steps.
  - Re-use comments for documentation.

University of Strathclyde Glasgow

Introducing Python

Visual studio code

https://code.visualstudio.com/



Debug

Extensions

Run

Edit

Output

Simple variables


```
box = 1
box = box + 1
print(box)
text = "The box"
text = text + " contains "
text += str(box)
print(text)
pi = 3.1459
r = 2
area = pi*r*r
print("Radius=" + str(r) + ", area=" + str(area))
```

Basic containers: Lists

University of Strathclyde Glasgow

```
numbers = [
    4,
    5,
    3
]
print("numbers =", numbers)
print("numbers[0] =", numbers[0])
print("numbers[-1] =", numbers[-1])
print("Each of the values:")
for number in numbers:
     print(number)
numbers = numbers[0:1] + [6] + numbers[1:]
```

Basic containers: Dictionaries

University of Strathclyde Glasgow

```
values = {
    "a": 5,
    "b": 6
}
values["c"] = 15
values["b"] = 10
values.update({
    "d": 20,
    "e": 9
})
print(values)
print(values["d"])
```

Comments

University of Strathclyde Glasgow

- Quickly understand code structure.
  - Comment on the intention, rather than the code syntax.
- Generate documentation from the comments.
- Comments have to be maintained too.
  - Limit number of comments used.

Comment types

University of Strathclyde Glasgow

- Single line comments using #.
- Multiple line comments using """ and """.

```
"""
A script to demonstrate Python comments,
which might span several lines.
"""

# Another print statement
print("Comment examples")    # This line prints a string
```

University of Strathclyde Glasgow