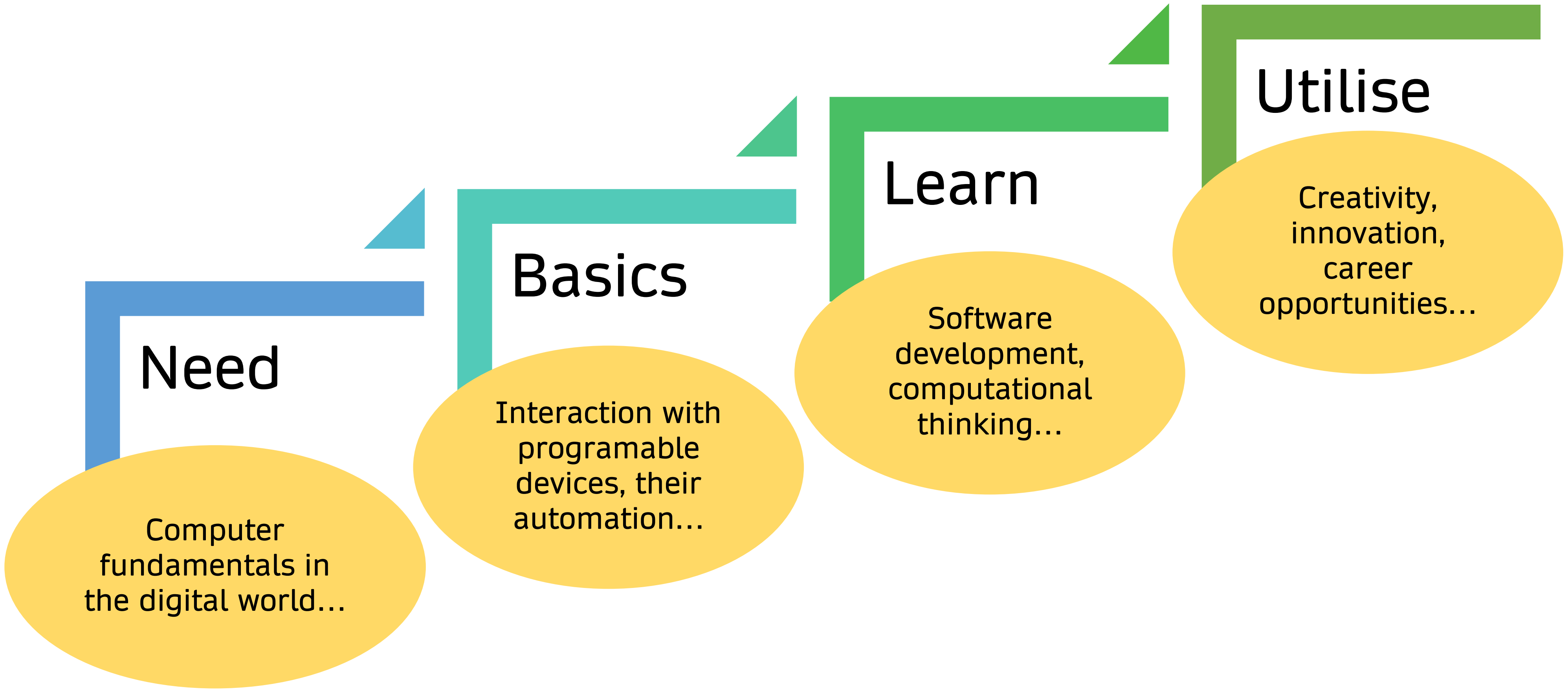
Computing (ES 112)

Yogesh K. Meena
Shouvick Mondal

August 2024

Computer Science & Engineering
IIT Gandhinagar

In this course…

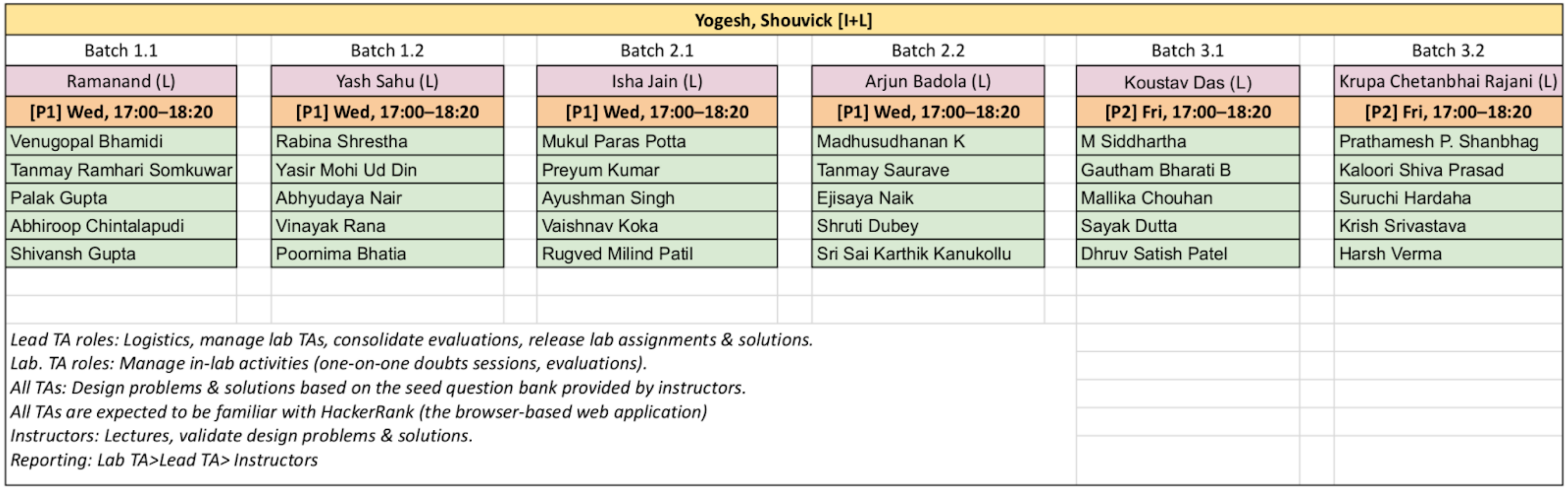

Need
Computer fundamentals in the digital world…
Basics
Interaction with programable devices, their automation…
Learn
Software development, computational thinking…
Utilise
Creativity, innovation, career opportunities…

Course Plan: Instructors and TAs

Instructors (02):

1. **Prof. Yogesh Kumar Meena** (doubt sessions: by appointment ONLY at AB13/401A)
2. **Prof. Shouvick Mondal** (doubt sessions: by appointment ONLY at AB13/402A)

TAs (36):



Yogesh, Shouvick [I+L]					
Batch 1.1	Batch 1.2	Batch 2.1	Batch 2.2	Batch 3.1	Batch 3.2
Ramanand (L)	Yash Sahu (L)	Isha Jain (L)	Arjun Badola (L)	Koustav Das (L)	Krupa Chetanbhai Rajani (L)
[P1] Wed, 17:00–18:20	**[P1] Wed, 17:00–18:20**	**[P1] Wed, 17:00–18:20**	**[P1] Wed, 17:00–18:20**	**[P2] Fri, 17:00–18:20**	**[P2] Fri, 17:00–18:20**
Venugopal Bhamidi	Rabina Shrestha	Mukul Paras Potta	Madhusudhanan K	M Siddhartha	Prathamesh P. Shanbhag
Tanmay Ramhari Somkuwar	Yasir Mohi Ud Din	Preyum Kumar	Tanmay Saurave	Gautham Bharati B	Kaloori Shiva Prasad
Palak Gupta	Abhyudaya Nair	Ayushman Singh	Ejisaya Naik	Mallika Chouhan	Suruchi Hardaha
Abhiroop Chintalapudi	Vinayak Rana	Vaishnav Koka	Shruti Dubey	Sayak Dutta	Krish Srivastava
Shivansh Gupta	Poornima Bhatia	Rugved Milind Patil	Sri Sai Karthik Kanukollu	Dhruv Satish Patel	Harsh Verma

Lead TA roles: Logistics, manage lab TAs, consolidate evaluations, release lab assignments & solutions.
Lab. TA roles: Manage in-lab activities (one-on-one doubts sessions, evaluations).
All TAs: Design problems & solutions based on the seed question bank provided by instructors.
All TAs are expected to be familiar with HackerRank (the browser-based web application)
Instructors: Lectures, validate design problems & solutions.
Reporting: Lab TA>Lead TA> Instructors

Course Plan: Contact Hours

Lectures:

- (Slot **F2**): Thu, 11:30–12:50 @ Jasubhai Auditorium

Lab sessions:

- (Slot **P1**): Wed, 17:00–18:20
  - Batch 1.1 (@ AB10/104)
  - Batch 1.2 (@ AB10/105)
  - Batch 2.1 (@ AB7/108)
  - Batch 2.2 (@ AB7/109)
- (Slot **P2**): Fri, 17:00–18:20
  - Batch 3.1 (@ AB10/104)
  - Batch 3.2 (@ AB10/105)

Overall course load:

{12L (x1)} + {12P (x2)}

Blue: Jasubhai (550x)

Red: AB10/104 (70x), AB10/105 (70x), AB7/108 (70x), AB7/109 (70x)

Course Plan: Timeline

Month	**Day**	Topics (slides)
August	Lec.: 22, 23(Thu), 29 Lab.: (21/?)~L1, (28/30)~L2	(Why program?)
September	Lec.: 5, 12, 19, 26 Lab.: (4/6)~L3, (11/13)~L4 (lab exam I), (?/20)~L5, (25/?)~L6	(Why program? contd.) (Variables, Expressions, and Statements) (Conditional Execution) (Functions)
October	Lec.: 17, 24 Lab.: (16/18)~L7, (23/25)~L8 (lab exam II), (30/*	(Loops and Iteration) (Strings) <Extra-class/Buffer> (Reading Files)
November	Lec.: 7, 14, 21 Lab.: */1)~L9, (6/8)~L10, (13/?)~L11, (20/22)~L12 (lab exam III)	(Lists) (Dictionaries) (Tuples)

Evaluation (relative grading)**:**
- [25%] Theory Exam I (Sep 27 – Oct 04, 2024)
- [25%] Theory Exam II (Nov 23 – 29, 2024)
- [15%] Lab Exam I (L4: Sep 11, 13)
- [15%] Lab Exam II (L8: Oct 23, 25)
- [20%] Lab Exam III (L12: Nov 20, 22)

Excluding holidays and breaks. **?** *indicates lab slots consumed by holidays/exams/breaks.*
Student Honour Code: [https://iitgn.ac.in/students/honourcode]

Study material to be followed

Texts/References:

- ✓ Python for Everybody, Charles Severance
- ✓ Python Crash Course, Eric Matthes
- ✓ Python Flash Cards (very useful for revisions), Eric Matthes
- ✓ The official Python tutorial and reference
- ✓ Dive Into Python 3, Mark Pilgrim
- ✓ Learn Python the hard way, Zed A Shaw
- ✓ A byte of Python, Swaroop CH
- ✓ Automate the boring stuff with Python, Al Sweigart

Lecture 1-2

Why program?

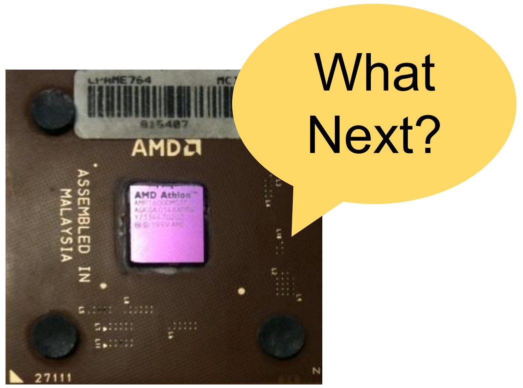
Lecture 1-2: Learning objectives

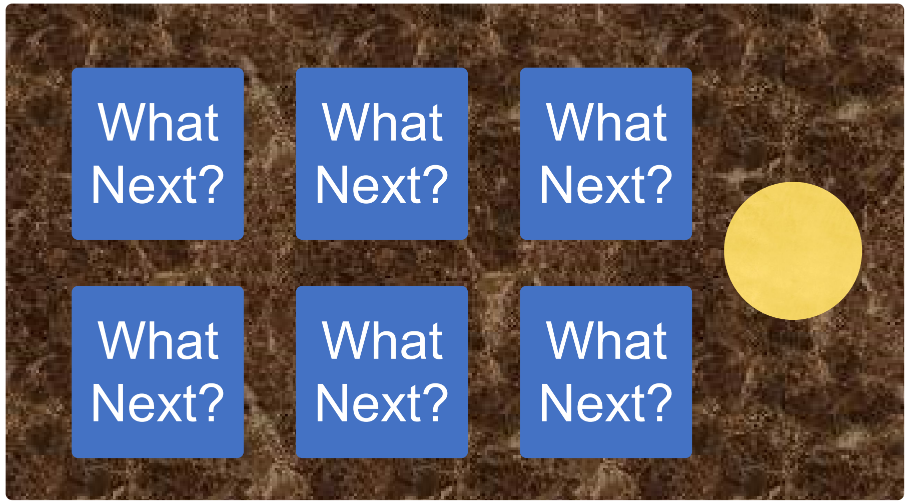
At the end of lecture 1-2 you should be able to:

- **Understand** the need for computation and programming
- **Recall** the importance of programing and its environment
- **Describe** the program structure and build on it
- **Summarize** why computer programming is needed

Computers Want to be Helpful…

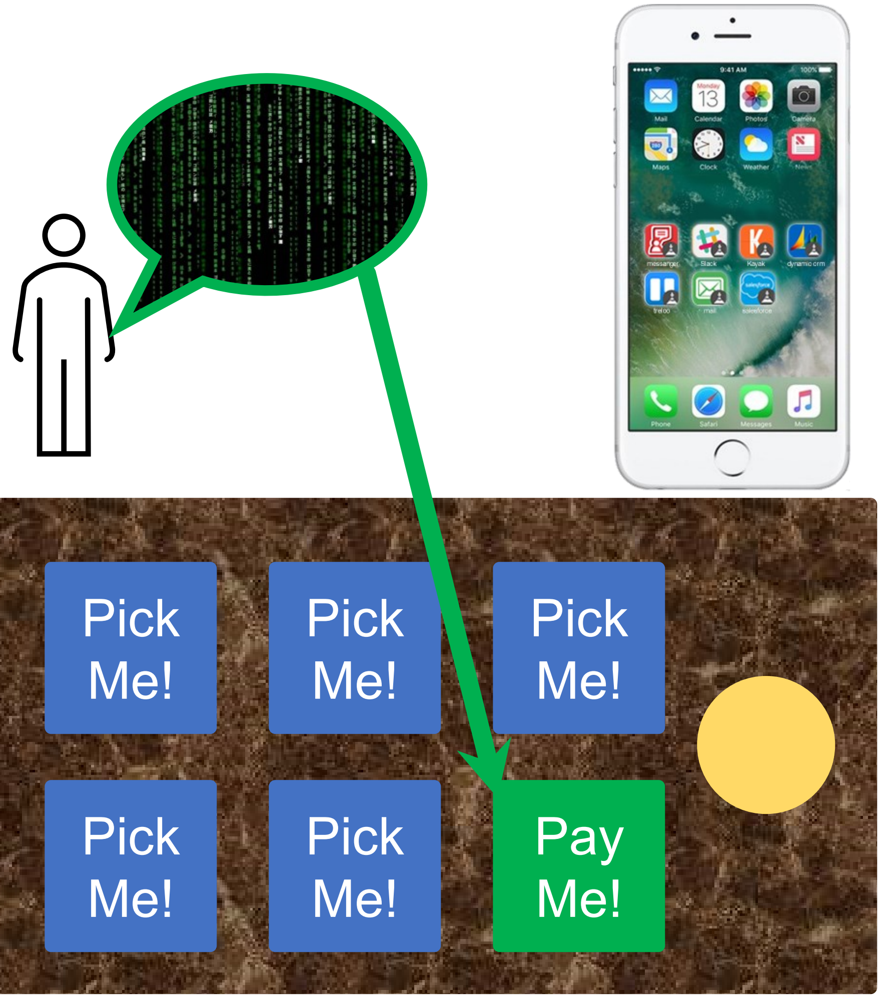
- Computers are built for one purpose - to do things for us
- But we need to speak their language to describe what we want done
- Users have it easy - someone already put many different programs (instructions) into the computer and users just pick the ones they want to use





Programmers Anticipate Needs

- iPhone applications are a market
- iPhone applications have over 3 billion downloads
- Programmers have left their jobs to be full-time iPhone developers
- Programmers know the ways of the program



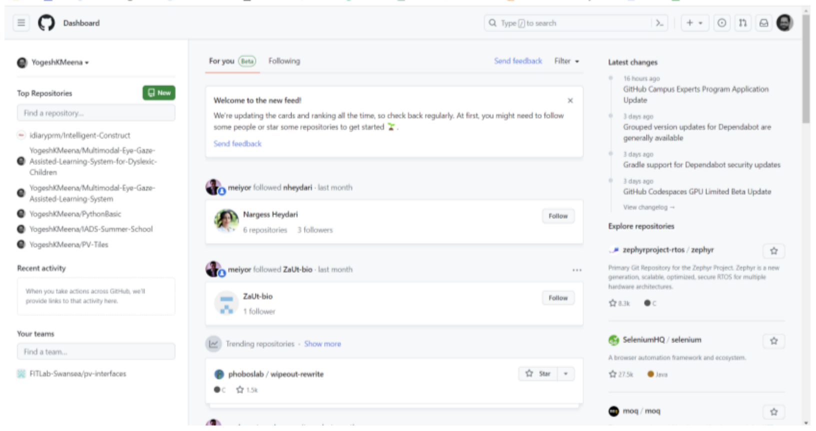
Users vs. Programmers

- Users see computers as a set of tools - word processor, spreadsheet, map, to-do list, etc.
- Programmers learn the computer “ways” and the computer language
- Programmers have some tools that allow them to build new tools
- Programmers sometimes write tools for lots of users and sometimes programmers write little “helpers” for themselves to automate a task

Why be a Programmer?

- To get some task done - we are the user and programmer
  - Clean up survey data
- To produce something for others to use - a programming job
  - Fix a performance problem in the Sakai software
  - Add a guestbook to a web site

Why be a Programmer?



Dashboard

Type / to search

YogeshKMeena

Top Repositories New

Find a repository...

idiarypm/Intelligent-Construct
YogeshKMeena/Multimodal-Eye-Gaze-Assisted-Learning-System-for-Dyslexic-Children
YogeshKMeena/Multimodal-Eye-Gaze-Assisted-Learning-System
YogeshKMeena/PythonBasic
YogeshKMeena/IADS-Summer-School
YogeshKMeena/PV-Tiles

Recent activity

When you take actions across GitHub, we'll provide links to that activity here.

Your teams

Find a team...

FITLab-Swansea/pv-interfaces

For you Beta Following Send feedback Filter

Welcome to the new feed!

We're updating the cards and ranking all the time, so check back regularly. At first, you might need to follow some people or star some repositories to get started 🌱.

Send feedback

meiyor followed nheydari · last month

Nargess Heydari
6 repositories 3 followers Follow

meiyor followed ZaUt-bio · last month

ZaUt-bio
1 follower Follow

Trending repositories · Show more

phoboslab / wipeout-rewrite Star
● C ☆ 1.5k

Latest changes

16 hours ago
GitHub Campus Experts Program Application Update

3 days ago
Grouped version updates for Dependabot are generally available

3 days ago
Gradle support for Dependabot security updates

3 days ago
GitHub Codespaces GPU Limited Beta Update

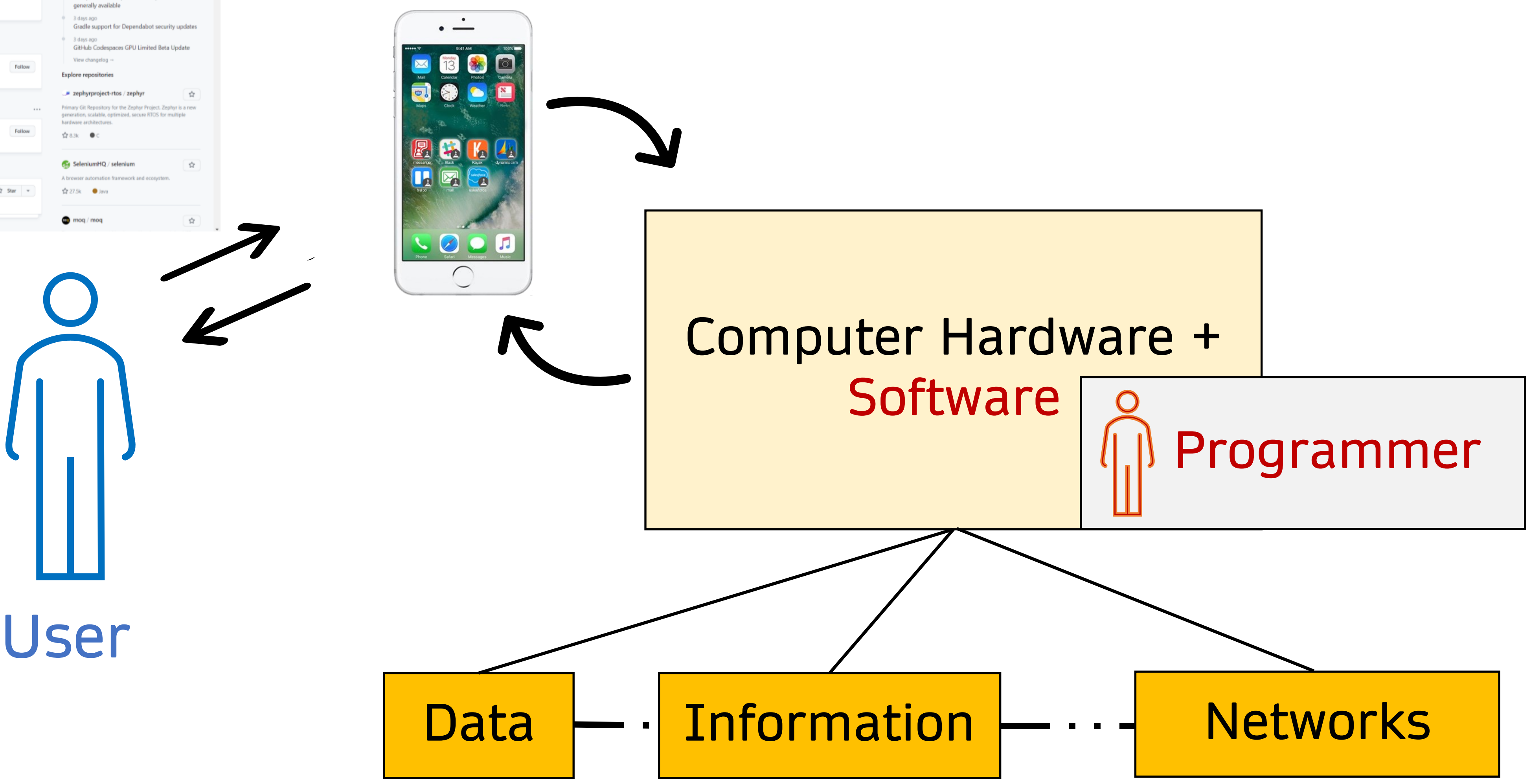
View changelog →

Explore repositories

zephyrproject-rtos / zephyr
Primary Git Repository for the Zephyr Project. Zephyr is a new generation, scalable, optimized, secure RTOS for multiple hardware architectures.
☆ 8.3k ● C

SeleniumHQ / selenium
A browser automation framework and ecosystem.
☆ 27.5k ● Java

moq / moq



What is Code? Software? A Program?

- A sequence of stored instructions
  - It is a little piece of our intelligence in the computer
  - We figure something out and then we encode it and then give it to someone else to save them the time and energy of figuring it out
- A piece of creative art - particularly when we do a good job on user experience

Programs for Humans…

```
while music is playing:
        Left hand out and up
        Right hand out and up
        Flip Left hand
        Flip Right hand
        Left hand to right shoulder
        Right hand to left shoulder
        Left hand to back of head
        Right ham to back of head
        Left hand to right hit
        Right hand to left hit
        Left hand on left bottom
        Right hand on right bottom
        Wiggle
        Wiggle
        Jump
```

https://www.youtube.com/watch?v=XiBYM6g8Tck

Programs for Humans...

```
while music is playing:
    Left hand out and up
    Right hand out and up
    Flip Left hand
    Flip Right hand
    Left hand to right shoulder
    Right hand to left shoulder
    Left hand to back of head
    Right ham to back of head
    Left hand to right hit
    Right hand to left hit
    Left hand on left bottom
    Right hand on right bottom
    Wiggle
    Wiggle
    Jump
```

https://www.youtube.com/watch?v=XiBYM6g8Tck

Programs for Humans…

```
while music is playing:
    Left hand out and up
    Right hand out and up
    Flip Left hand
    Flip Right hand
    Left hand to right shoulder
    Right hand to left shoulder
    Left hand to back of head
    Right hand to back of head
    Left hand to right hip
    Right hand to left hip
    Left hand on left bottom
    Right hand on right bottom
    Wiggle
    Wiggle
    Jump
```

https://www.youtube.com/watch?v=vlzwuFkn88U

Programs for Python...

the clown ran after the car and the car ran into the tent and the tent fell down on the clown and the car

Which is the most frequently occurring word here?

Image: https://www.flickr.com/photos/allan_harris/4908070612/ Attribution-NoDerivs 2.0 Generic (CC BY-ND 2.0)

Python code to count the most frequent word

```
name = input('Enter file:')
handle = open(name)

counts = dict()
for line in handle:
    words = line.split()
    for word in words:
        counts[word] = counts.get(word,0) + 1

bigcount = None
bigword = None
for word,count in counts.items():
    if bigcount is None or count > bigcount:
        bigword = word
        bigcount = count

print(bigword, bigcount)
```

```
python words.py
Enter file: clown.txt
the 7
```

the clown ran after the car and the car ran into the tent and the tent fell down on the clown and the car

Python code to count the most frequent word

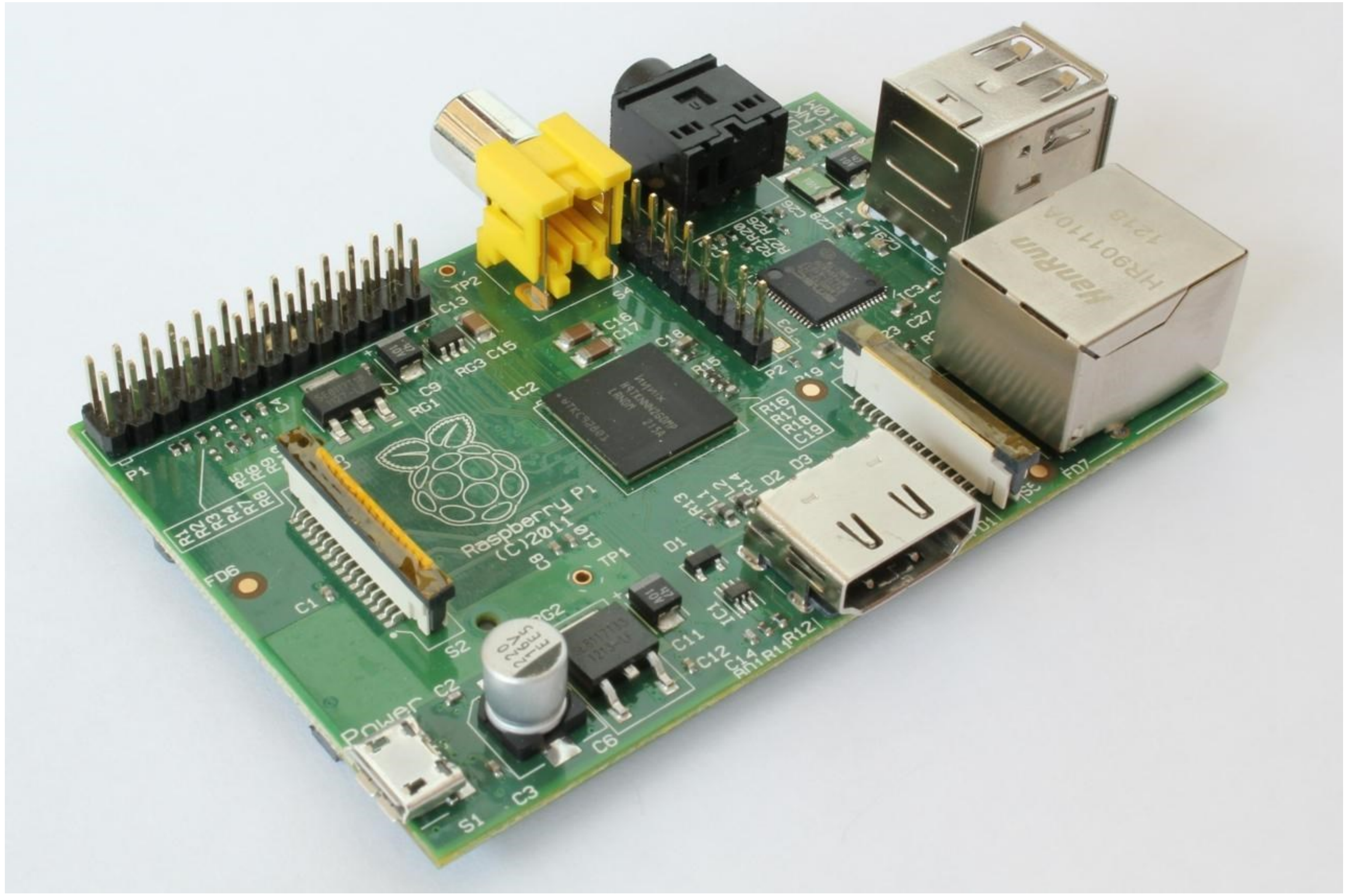
Writing programs or programming is a very creative and rewarding activity You can write programs for many reasons ranging from making your living to solving a difficult data analysis problem to having fun to helping someone else solve a problem This book assumes that {\em everyone} needs to know how to program and that once you know how to program, you will figure out what you want to do with your newfound skills

We are surrounded in our daily lives with computers ranging from laptops to cell phones We can think of these computers as our personal assistants who can take care of many things on our behalf The hardware in our current-day computers is essentially built to continuously ask us the question What would you like me to do next

Our computers are fast and have vasts amounts of memory and could be very helpful to us if we only knew the language to speak to explain to the computer what we would like it to do next If we knew this language we could tell the computer to do tasks on our behalf that were reptitive Interestingly, the kinds of things computers can do best are often the kinds of things that we humans find boring and mind-numbing

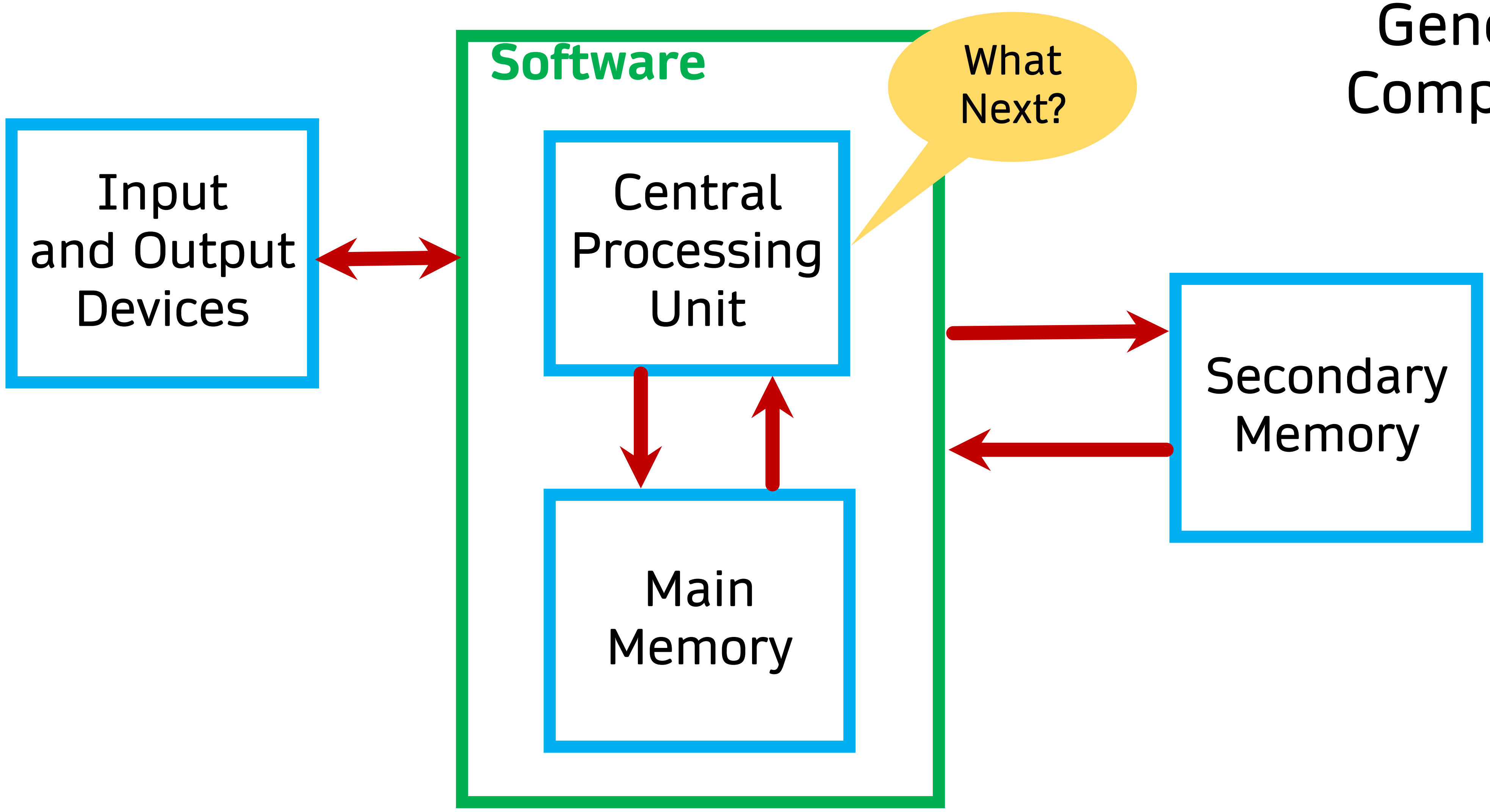
```
python words.py
Enter file: words.txt
to 16
```

Hardware Architecture



http://upload.wikimedia.org/wikipedia/commons/3/3d/RaspberryPi.jpg

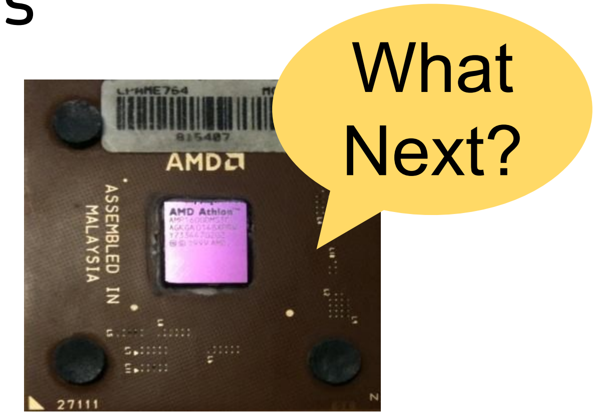
Hardware Architecture



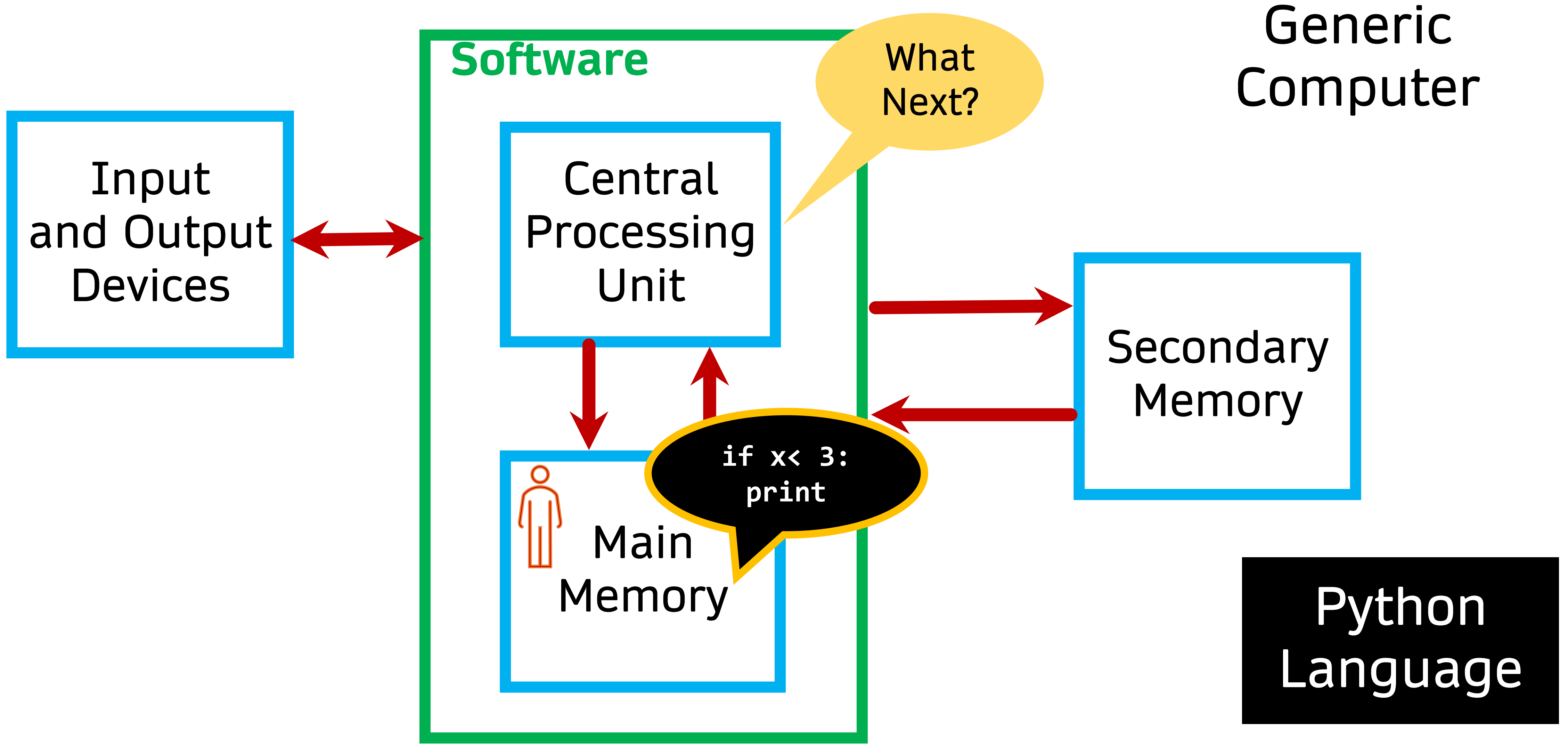
Generic Computer

Hardware Architecture: Definitions

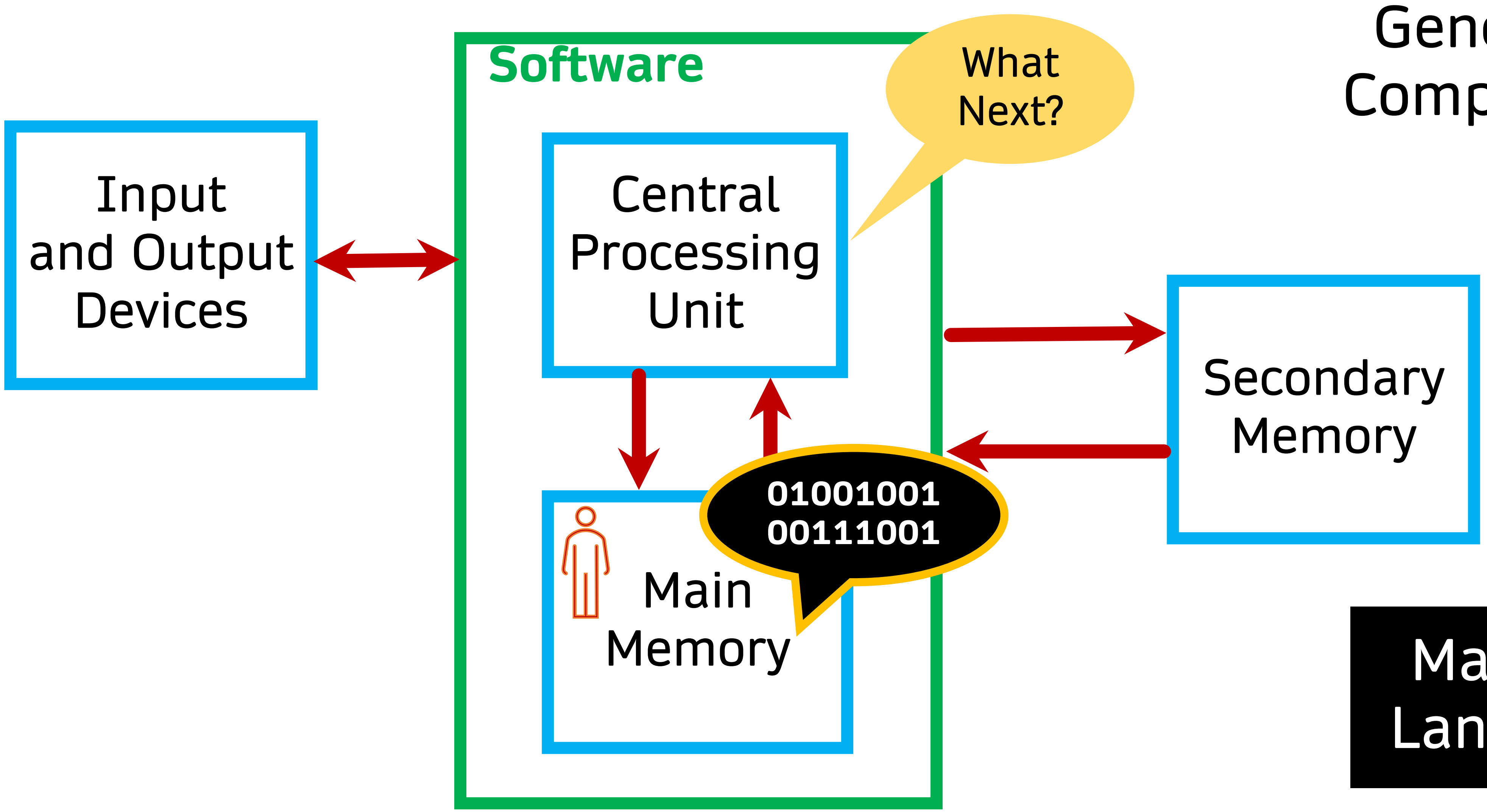
- Central Processing Unit: Runs the Program - The CPU is always wondering "what to do next". Not the brains exactly - very dumb but very very fast
- Input Devices: Keyboard, Mouse, Touch Screen
- Output Devices: Screen, Speakers, Printer, DVD Burner
- Main Memory: Fast small temporary storage - lost on reboot - aka RAM
- Secondary Memory: Slower large permanent storage - lasts until deleted - disk drive / memory stick



Hardware Architecture



Hardware Architecture



Generic Computer

Machine Language

Hardware in action: Totally Hot CPU

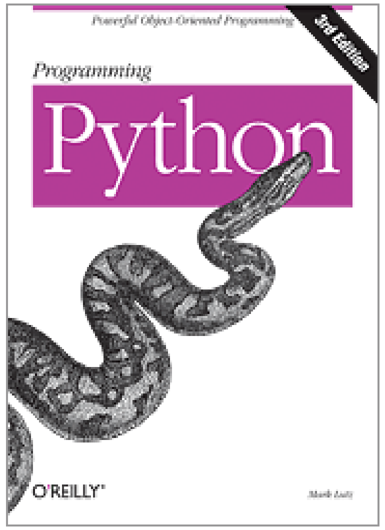
[What happens when the CPU cooler is removed?]

Hardware in action: Hard Disk

Inside of Hard Drive

Python as a Language

- **Python** is the language of the Python Interpreter and those who can converse with it.
- An individual who can speak **Python** is known as a Pythonista. It is a very uncommon skill and may be hereditary.
- Nearly all known Pythonistas use software initially developed by **Guido van Rossum**.



What is an Interpreter?

A computer program that **directly executes instructions** written in a **programming** language, **without requiring prior translation** into a machine language program.

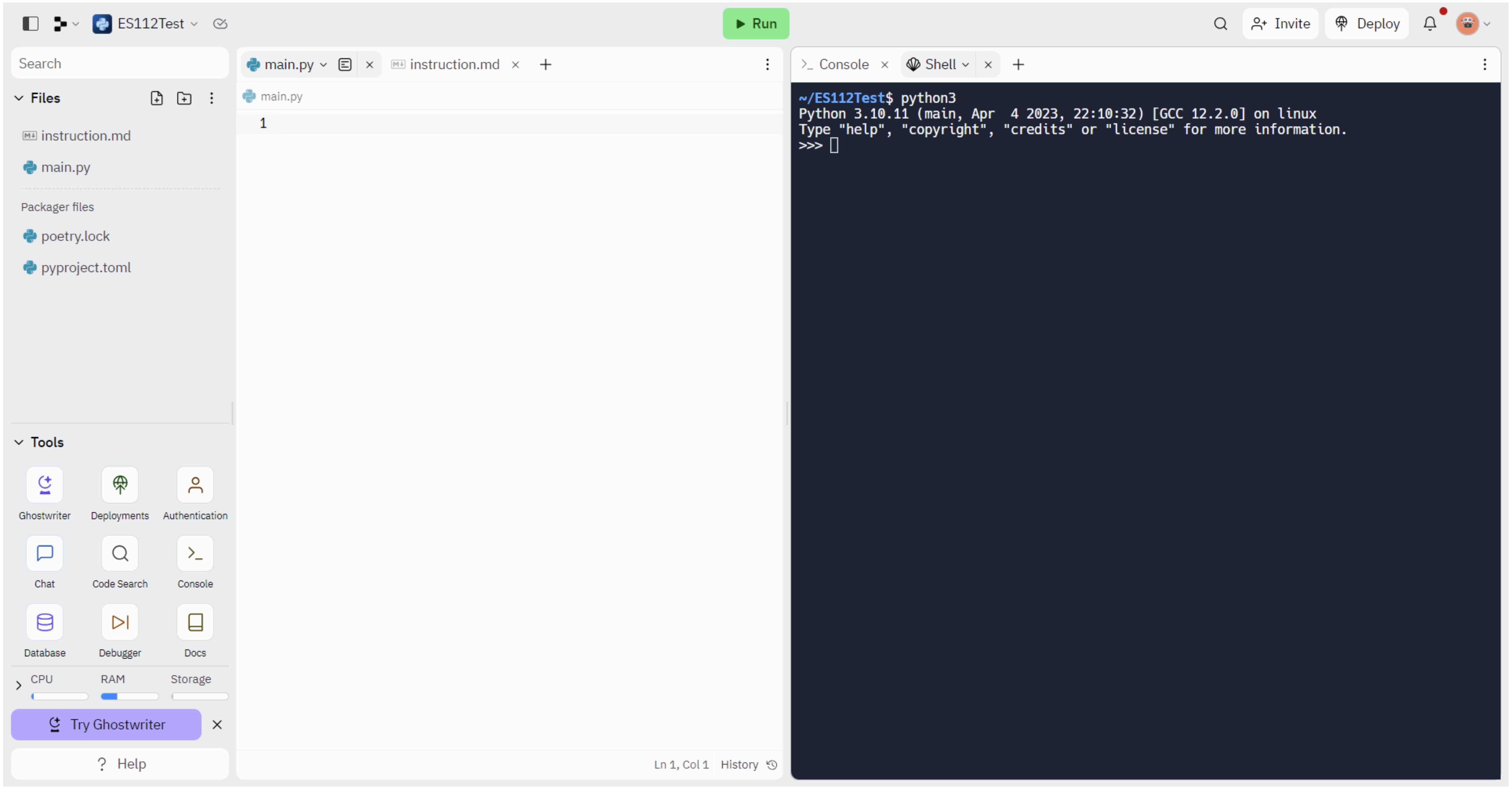
An interpreter generally uses **one** of the following strategies for program execution:

- **Parse** the source code and **perform its behavior** directly;
- **Translate** source code **into** some efficient **intermediate representation** or object code and **immediately execute** that;
- Explicitly **execute stored precompiled bytecode** made by a compiler and matched with the interpreter **Virtual Machine**.

Early Learner: Syntax Errors

- We need to learn the **Python language** so we can communicate our instructions to Python. In the beginning we will make lots of mistakes and speak gibberish like small children.
- When you make a mistake, the computer does not think you are “cute”. It says “**syntax error**” - given that it knows the language and you are just learning it. It seems like Python is cruel and unfeeling.
- You must remember that you are intelligent and can learn. The computer is simple and very fast but cannot learn. So, **it is easier for you to learn Python than for the computer to learn English…**

Talking to Python Interpreter (Shell mode)



```
~/ES112Test$ python3
Python 3.10.11 (main, Apr  4 2023, 22:10:32) [GCC 12.2.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> 
```

Talking to Python

```
~/ES112Test$ python3
Python 3.10.11 (main, Apr  4 2023, 22:10:32) [GCC 12.2.0] on linux
Type "help", "copyright", "credits" or "license" for more
 information.
>>>
```

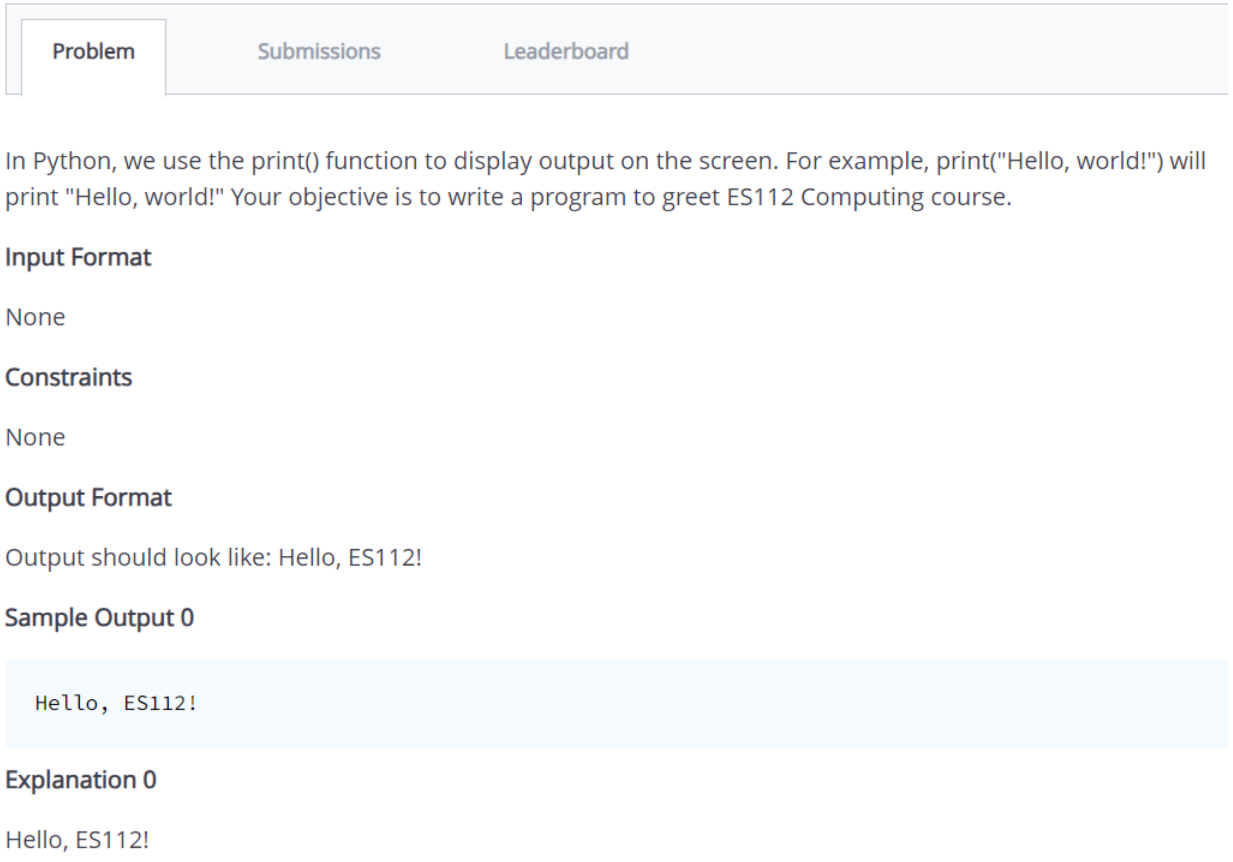
What Next?

Talking to Python

```
~/ES112Test$ python3
Python 3.10.11 (main, Apr  4 2023, 22:10:32) [GCC 12.2.0] on linux
Type "help", "copyright", "credits" or "license" for more
 information.
>>> print("Hello World")
Hello World
>>>
```

What Next?

Talking to Python (HackerRank Platform)



In Python, we use the print() function to display output on the screen. For example, print("Hello, world!") will print "Hello, world!" Your objective is to write a program to greet ES112 Computing course.

Input Format

None

Constraints

None

Output Format

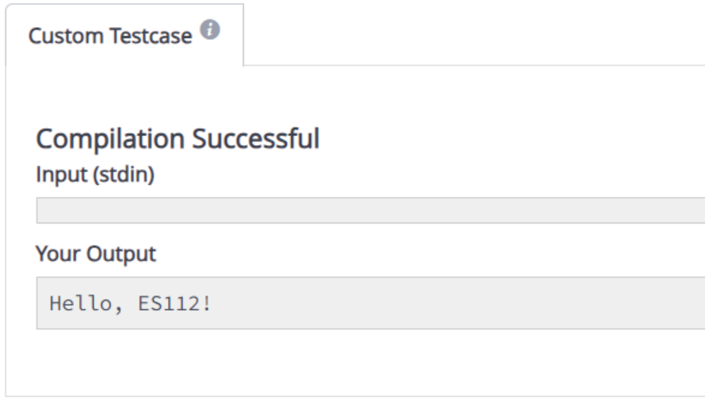
Output should look like: Hello, ES112!

Sample Output 0

```
Hello, ES112!
```

Explanation 0

Hello, ES112!

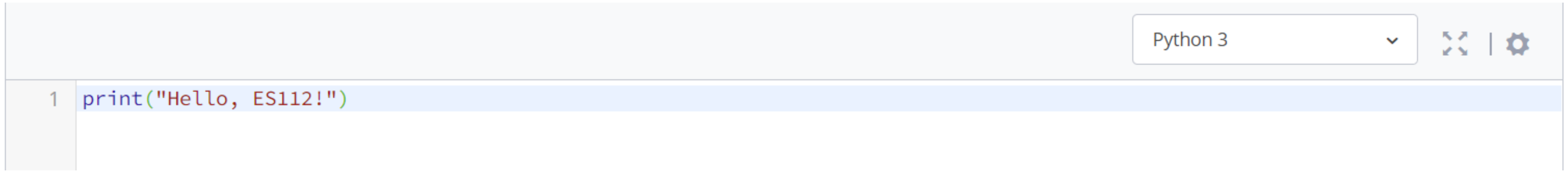


Compilation Successful

Input (stdin)

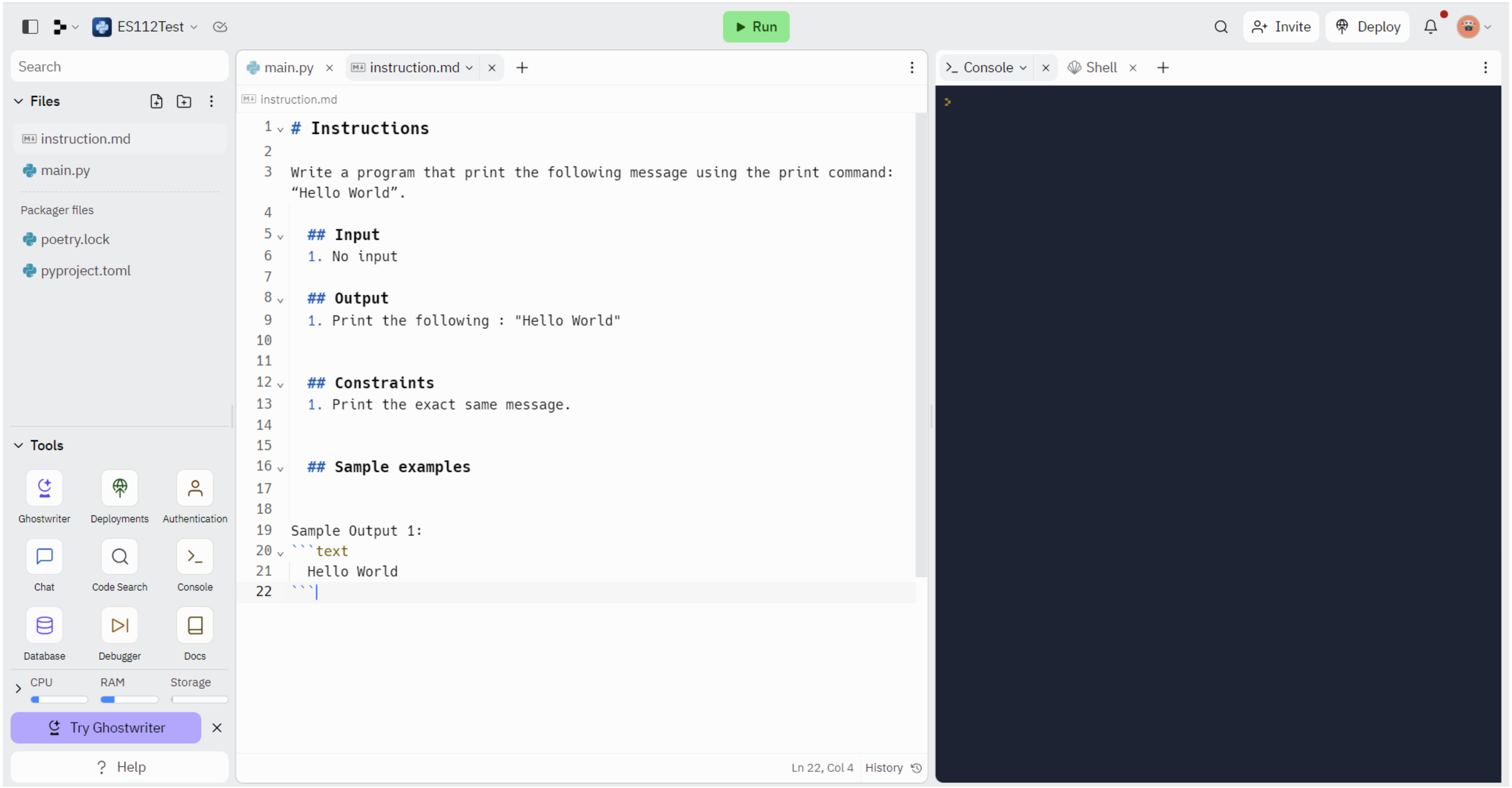
Your Output

```
Hello, ES112!
```



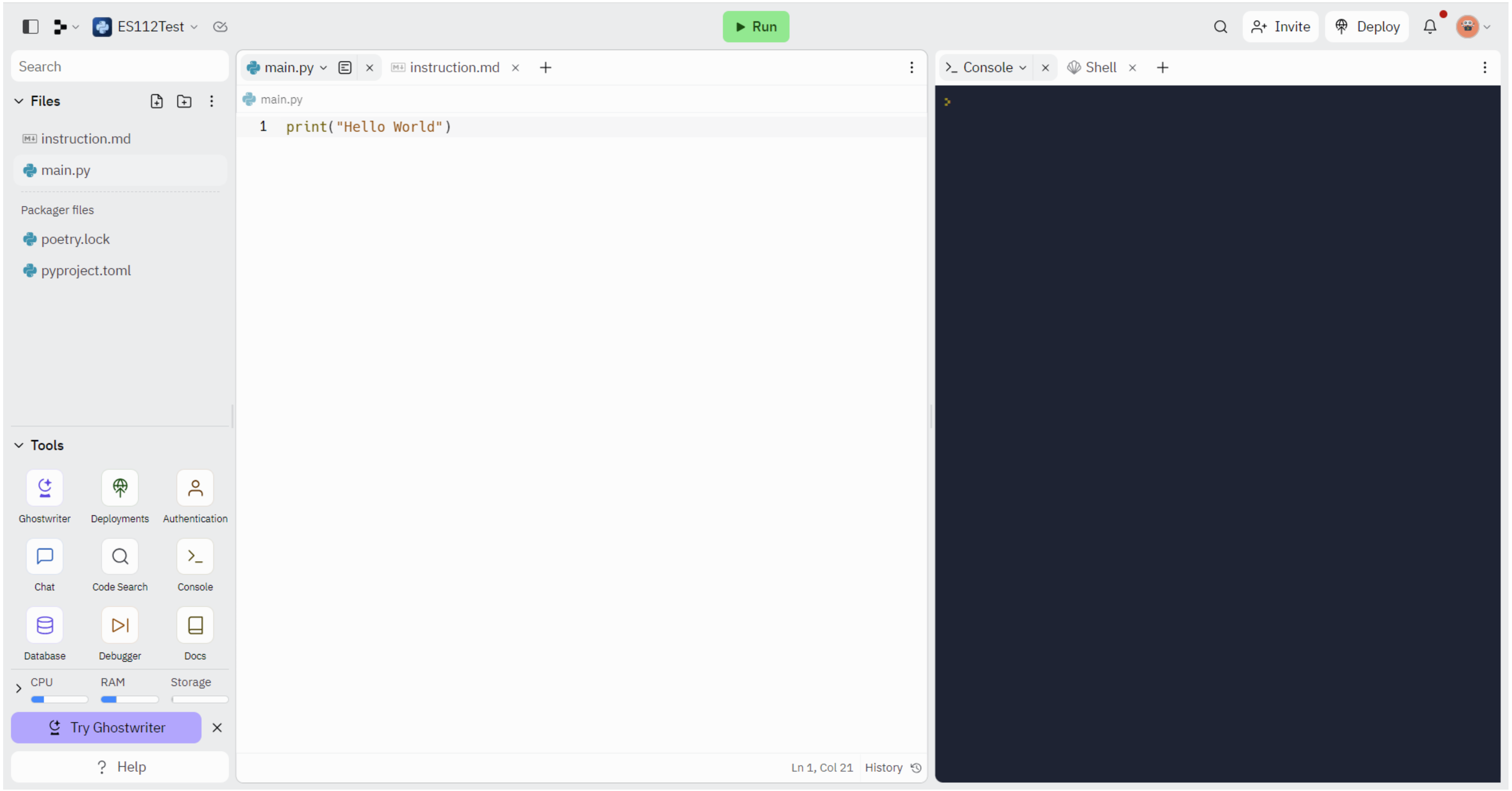
```python
print("Hello, ES112!")
```

Talking to Python (Replit Platform)



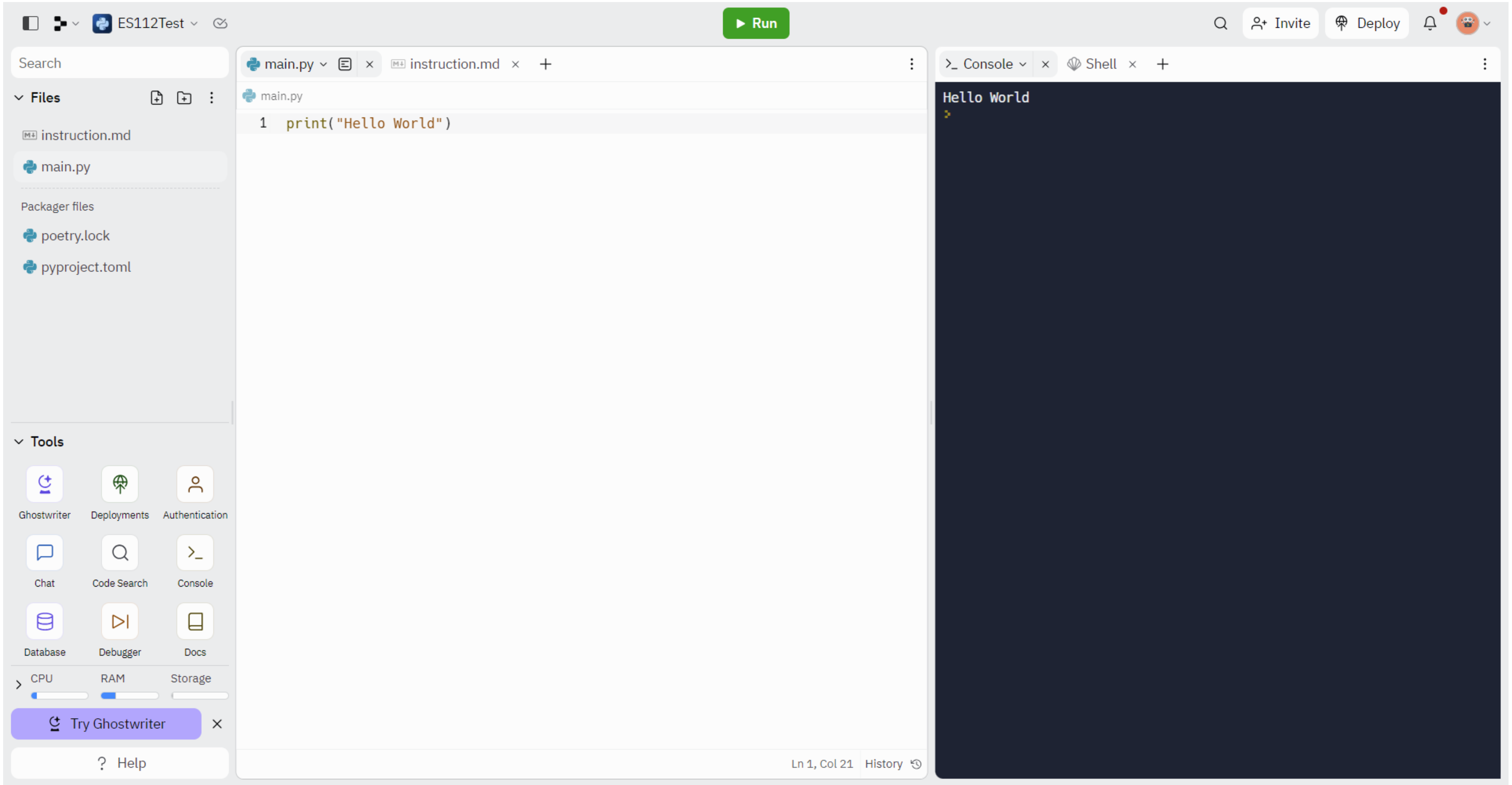
~~~
# Instructions

Write a program that print the following message using the print command:
“Hello World”.

 ## Input
 1. No input

 ## Output
 1. Print the following : "Hello World"


 ## Constraints
 1. Print the exact same message.


 ## Sample examples


Sample Output 1:
```text
 Hello World
```
~~~

Talking to Python (Replit Platform)



```python
print("Hello World")
```

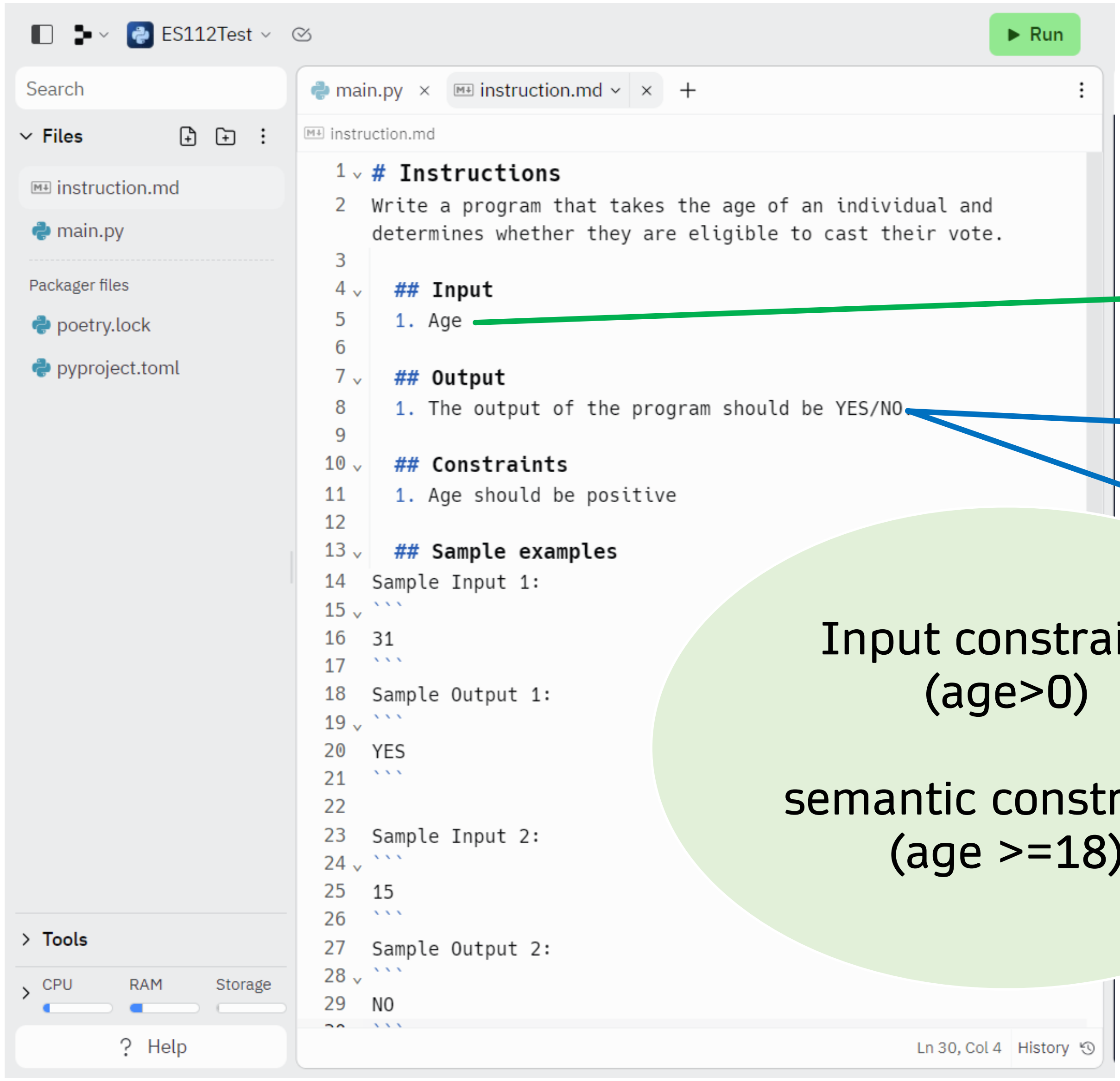
Talking to Python (Replit Platform)



```
print("Hello World")
```

```
Hello World
```

Workflow of program construction



```
# Instructions
Write a program that takes the age of an individual and
determines whether they are eligible to cast their vote.

  ## Input
  1. Age

  ## Output
  1. The output of the program should be YES/NO

  ## Constraints
  1. Age should be positive

  ## Sample examples
Sample Input 1:
```
31
```
Sample Output 1:
```
YES
```

Sample Input 2:
```
15
```
Sample Output 2:
```
NO
```
```

Program

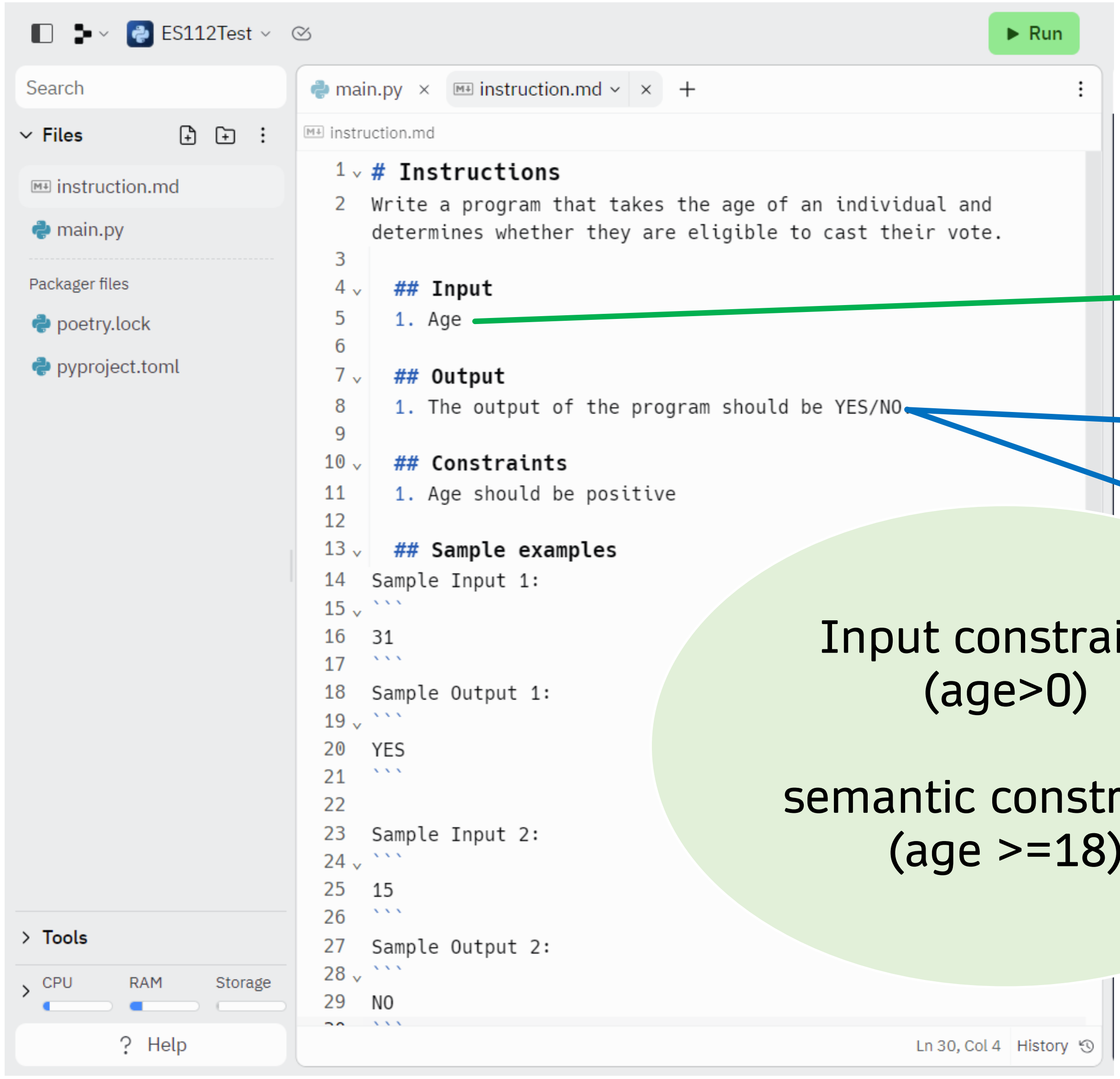
```
age=input()
if age >= 18:
    print("YES")
else:
    print('NO')
```

Input constraints
(age>0)

semantic constraints
(age >=18)

Will this code run?

Workflow of program construction



```
# Instructions
Write a program that takes the age of an individual and
determines whether they are eligible to cast their vote.

 ## Input
 1. Age

 ## Output
 1. The output of the program should be YES/NO

 ## Constraints
 1. Age should be positive

 ## Sample examples
Sample Input 1:
```
31
```
Sample Output 1:
```
YES
```

Sample Input 2:
```
15
```
Sample Output 2:
```
NO
```
```

Input constraints (age>0)

semantic constraints (age >=18)

Program

```
age=int(input())
if age >= 18:
    print("YES")
else:
    print('NO')
```

Now, will this code run?

Talking to Python Interpreter (Shell mode)

```
~/ES112Test$ python3
Python 3.10.11 (main, Apr  4 2023, 22:10:32) [GCC 12.2.0] on linux
Type "help", "copyright", "credits" or "license" for
 more information.
>>> x = 1
>>> print(x)
1
>>> x = x + 1
>>> print(x)
2
>>> exit()
```

This is a good test to make sure that you have Python correctly working. Note that quit() also works to end the interactive session.

Acknowledgements / Contributions

Initial Development: Charles Severance, University of Michigan School of Information

Contributors 2024 - Yogesh K. Meena and Shouvick Mondal, IIT Gandhinagar