CS 370

Producer and Consumer, Synchronization

Assignment Review

- You are supposed to implement a solution to the Producer and Consumer problem, using a circular FIFO buffer.
- There will be at least two Consumers and at least two Producers.
- **Producers**: are supposed to generate a certain number of characters. The Characters must be randomly generated between 'a' and 'z' (both inclusive). It also keeps track of all the characters produced by storing them in a String.
- **Consumers**: are supposed to consume the elements, produced by the Producers. Each consumer will keep its own String containing the elements it has consumed.
- Both, Producers and all Consumers, are supposed to report the character produced/consumed along with the index and timestamp with nanosecond resolution.

Producer.java

- The Producers will produce the total number of elements which will be passed as the second argument.
- An identification is passed as the third argument (begin with 1 and increment) to identify each Producer.
- A seed is used to set the random generator to generate the same sequence every time the same seed is given. It is passed as the fourth argument.
- Generate a character between 'a' and 'z' (both inclusive) and insert it into the buffer.
- A producer cannot insert an element into the buffer when the buffer is full
- If a number is inserted successfully, it is appended to the String to keep track of all the generated characters.

Consumer.java

- A Consumer consumes an element from the buffer.
- Each Consumer will consume a ratio of the total elements (number of elements / number of consumers) if it is evenly divisible.
- A Consumer cannot consume an element when the buffer is empty.
- Once a Consumer consumes an element from the buffer successfully, it adds that character to the String to keep track of the consumed elements.

Bdbuffer.java

- `Bdbuffer.java` contains the circular FIFO buffer that will be used among all the producers and the consumers
- It also has the required functions that is used to insert or remove an element, and it returns the appropriate values.
- It may additionally have other functions such as isFull(), isEmpty(), etc. depending on your implementation.

Invoker.java

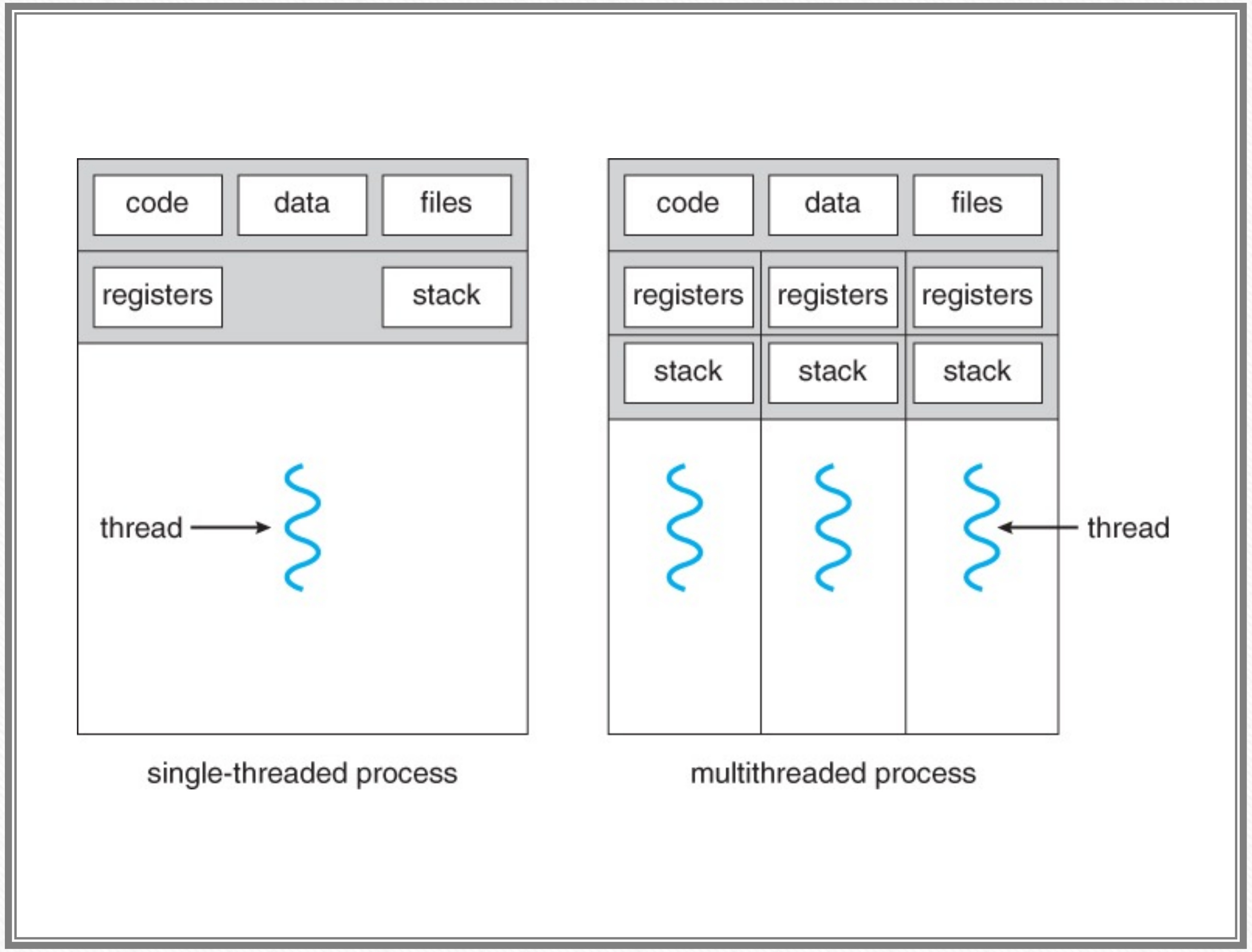
- It creates one instance of the buffer, creates required number of threads of producers, creates required number of threads of consumers, and then waits for all of them to finish.
- Once all threads terminate, we get the Strings generated by each of the Producers and the Strings generated by each of the consumers.
- Essentially, all the produced elements must be consumed. However, they may be out of order. Hence, we sort both the strings generated/consumed and check if they are the same.

Synchronization in Java

- Java has inbuilt monitors
    - Allows threads to have mutual exclusion
    - Allows threads the ability to wait (block) for a condition to become true
- Signalling is done using
    - wait()
    - notify() or notifyAll()

- Built in thread class can be extended and used
    - Instantiate and use myThread.start()
    - @Override run() to change what a thread does

Threads



single-threaded process

multithreaded process

```
public class PhilosopherThread
extends Thread
{
        @Override
        public void run()
        {
                // Thread entry point
        }
}
```

Creating and Starting threads

```
public class PhilosopherThread extends Thread {
    @Override
    public void run()  {
        // Thread entry point
    }
}
```

```
PhilosopherThread Socrates = new PhilosopherThread(table, seat);

Socrates.start(); //begins Socrates thread invokes the run() method
```

Synchronized methods

- A piece of logic marked with synchronized becomes a synchronized block, allowing only one thread to execute at any given time.

```
public synchronized void pickup(int i) throws InterruptedException
{
    //Synchronized code goes in here
}
```

wait(), notify() and notifyAll()

- wait()
  - Causes current thread to wait until another thread invokes the notify() or notifyAll() method
- notify()
  - notify() wakes up one thread waiting for the lock
- notifyAll()
  - The notifyAll() method wakes up all the threads waiting for the lock; the JVM selects one of the threads from the list of threads waiting for the lock and wakes that thread up

CS 370

Raspberry Pi

Topics

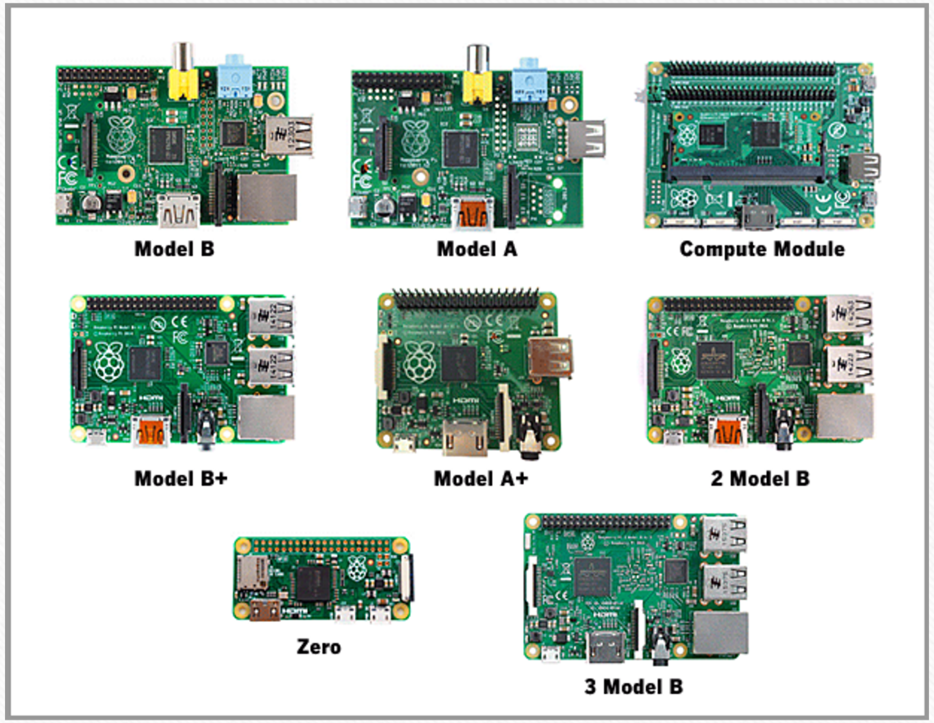
- Intro to Raspberry Pi
- Setting up a Raspberry Pi
- Term Project Requirements
- Term Project Expectations
- Helpful Links

Why Raspberry Pi's

- Small and Portable
- Cheap
- Well-Documented
- Versatile
- Support for many peripherals (thanks to Linux)

Third Best Selling Computer Brand in the World

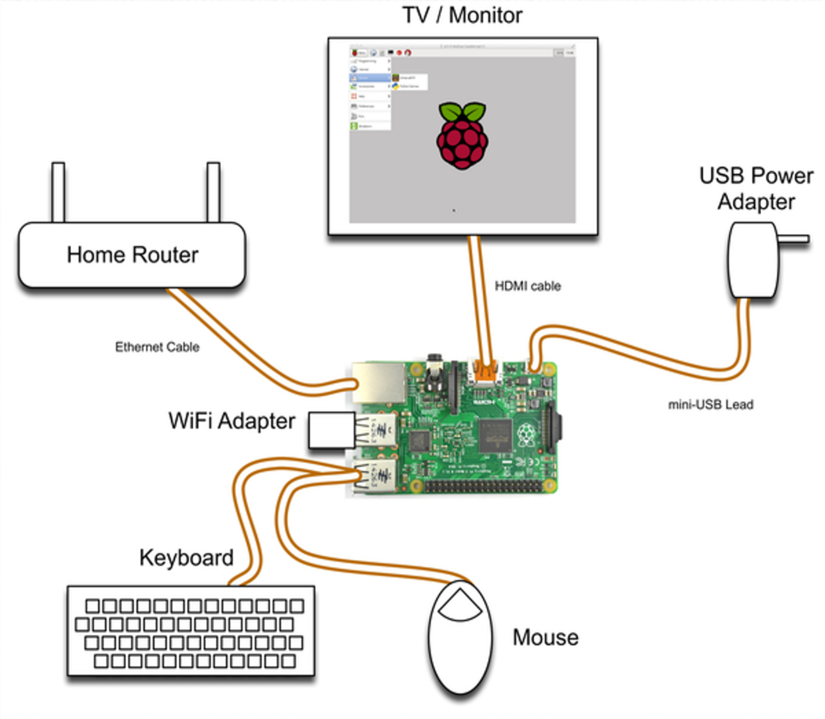
Raspberry Pi Models



Raspberry Pi 3 Model B+

- 1.4GHz 64-bit quad-core processor
- dual-band wireless LAN
- Bluetooth 4.2/BLE
- faster Ethernet
- Power-over-Ethernet support (with separate PoE HAT)
- Raspberry Pi 4 - Even more memory

Raspberry Pi Setup

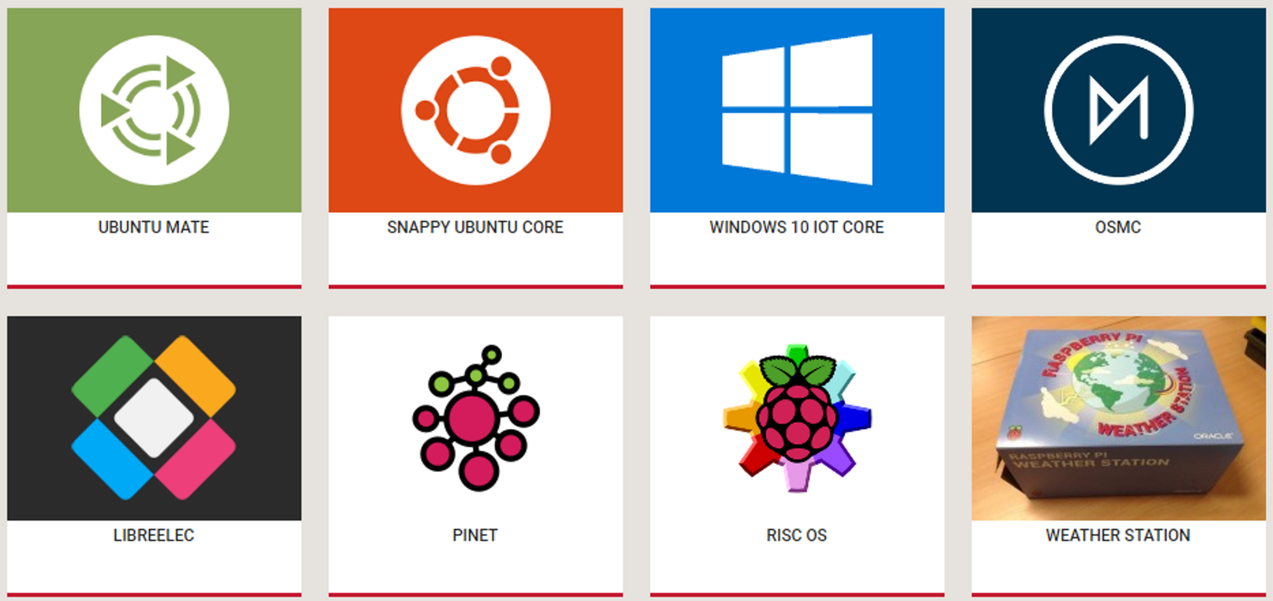


Can connect to monitor, keyboard, mouse

Usable as a normal desktop

Optionally use *ssh* instead of a monitor

Raspberry Pi Operating Systems



Expect most groups to use Raspbian (officially supported OS)

Other options are available - some OS's for specific use cases

Programming Languages

Basically any language will work (Python, C, Java, C++, Javascript, Ruby, Lisp, Rust, R, etc…)

Most projects done in Python or C

GPIO Libraries

Python/C

- RPi.GPIO (Python)
  - RPi.GPIO code samples
- RPIO.GPIO (Python)
- wiringPi (Python/C)
- pigpio (Python/C/Javascript)
- gpiozero (Python)
- bcm2835 (C)

Term Project Requirements

Project must involve:

- A single board computer (Raspberry Pi)
  - With WiFi capability + operating system
- Communication with at least one other computer
  - Another board, desktop, assistant, etc.
- At least one sensing or interacting device
  - Heat sensor, motion detector, camera, motor, controller, etc...

Term Project TODO

- Team Composition and Proposal (done – 5%)
- Progress Report (due on 11/4/2021 - 15%)
- Final Report and Demo
  - Report: 1500 - 2500 words
  - Code
  - 10 - 15 Minute Demo
- Presentation
- Peer Review (5%)

Term Project Expectations

- Originality
    - Several groups with similar projects (temperature sensors, plant waterers, etc...)
    - Come up with a unique selling point
        - Find similar projects online, then do something different
- Thoroughness
    - Think about the evaluations you're performing - design careful experiments and control for variables
    - Try to learn something you couldn't have guessed

Helpful Links

- Help Guides
  - Setup instructions
  - SSH with Raspberry Pi's
  - Help videos
  - FAQ's
  - Embedded Linux wiki

- Forums and Tutorials
  - Raspberry Pi forums / projects
  - Hackaday Projects
  - Adafruit Learning Guides
  - Raspberry Pi subreddit

Thank You

Questions?