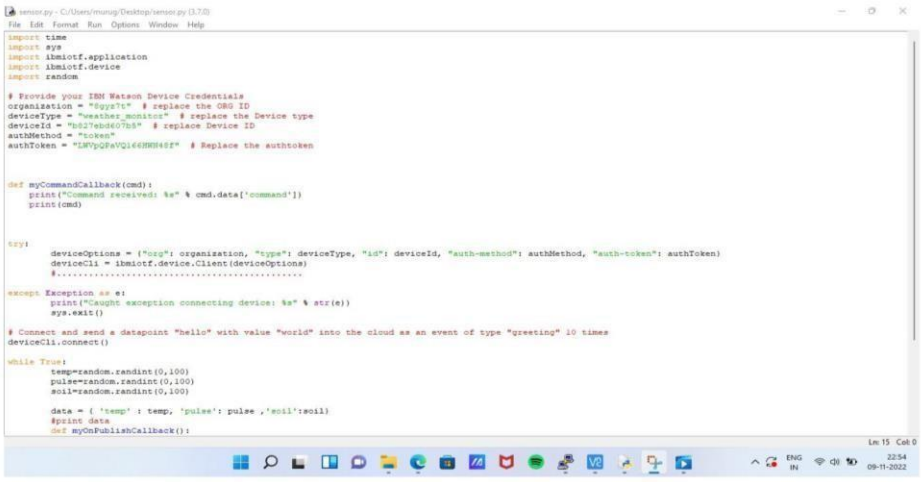
| | |
|---|---|
| **Date** | **16 November 2022** |
| **Team ID** | **PNT2022TMID20342** |
| **Project** | **IoT Based Smart Crop Protection System For Agriculture** |

# Coding layout,readability ,reusability

## Coding layout

**STEP 1**: Write a python code for randomize Soil Moisture ,Temperature and Humidity.



# code readability

Readable code is simply code that clearly communicates its intent to the reader. Most likely, the code we write will be read by other developers, who will either want to understand or modify the way our code works. They may need to test the code, fix a problem, or add a new feature.

These other developers reading our code could be our teammates, a consultant, a new junior developer, or even developers from another company who are programming a specific portion of the application. And more importantly, that developer could be ourselves, when we revisit our own code. With most languages, the ratio of time spent reading vs. writing is well over 10:1.

# Why should we produce readable code?

Since our software code can be shared with other developers, we'll want to make it easier to work with this shared code. To do this, we'll have to make sure that everyone involved, including ourselves, understands the same thing easily and quickly. Code that is readable for one person is not necessarily readable for another.

Code that is not readable takes more time to understand and can make us lose a lot of time on what should be a simple task. The worst-case scenario is that it could even take several iterations to fix some problems. In some cases, we might spend so much time trying to understand code that we might want to rewrite it completely.

Poorly written code would even make refactoring difficult. When rewriting poorly written code, by removing the original code and writing a new implementation, we might not cover all the use cases of that code, especially if the documentation of the requirements is limited or absent. As a result, the time spent rewriting will be greater than the time it took to write the original code.

Furthermore, by not understanding the code well, we may misinterpret its use and by modifying the code, we may unintentionally create new problems. Code that is not easily readable can therefore increase the risk of defects.

# Benefits of Creating Readable Code Benefits of Creating Readable Code

On the other hand, readable and well-tested code is what makes it easier to refactor, extend and modify parts of the system because it is easier and less time-consuming to understand. Readable and well-tested code is the foundation of a solid base, where developers are confident and quick to make changes.

Good readable code obviously takes more effort to produce than quickly written and poorly planned code. At Direct Impact, we believe that the benefits of readable code win over quickly produced code that may contain more errors.

In order to produce good readable code, we will have to follow good coding practices.

## How do we make our code more readable?

It would be easy to say that by simply adding comments to our code, it will automatically become more readable. But readable code involves much more. Adding relevant comments everywhere in our code would be a very laborious task and could become overly commented code. Overly commented code would have the opposite effect, making the code harder to read and understand.

# REUSABILITY

abstract

This paper presents a knowledge-based approach for selecting and testing modular reusable code. This approach entails three stages: system definition through a system entity structure (SES), SES pruning and model synthesis using an expert system (ES), and the evaluation of candidate design models using discrete event simulation (DEVS). An example of this approach is shown through the development of a Group Decision Support System (GDSS) idea generation tool.

Algorithm 1: STARVATION REMOVAL ALGORITHM

```
if scheduling lease type= = Immediate lease then:
        if resources required are available at the time then:
                allocate resources to Immediate lease
        else
                reject Immediate lease and print message
else
        if scheduling lease type= = BE then:
                Queue BE lease and set state of lease to queue.
        else
                if scheduling lease type= =AR then:
                        if BE lease request already scheduled at that time then:
                         if number of suspension > max limit provided by user in
                         advance for that BE then:
                                Do not schedule this AR & reschedule BE to be
                                suspended on slot vacated by AR
                         else
                                Increase counter of suspension of that BE and
                                allocate resources to AR lease
                        else
                         if resources are not available as being used by other AR
                         lease request then:
                                reject the AR lease and print message
                else
                        if scheduling lease type= = DLS then:
                                slack=(deadline-start time) / duration
                                if slack < 1.1 then
                                 reject lease and print message to extend dead line or
                                 submit lease as AR lease
                                else
                                 find a single time slot which can satisfy complete
                                 lease within deadline
                                 if new lease is not schedulable as above then:
                                        find multiple slots which together can satisfy
                                        this conditions
                                 if new lease is not schedulable by any of the above
                                 methods then:
                                   find leases to be rescheduled & reschedule
                                   deadline leases
                                 else
                                   reject new lease & print message
                        else
                                print message invalid lease type
```