# Communicating Computer Science in Schools Final Report

Jack Morrison
*CID:01343274*

March 2020

## Contents

## Abstract

From the 14th of January until the 5th of March 2020, I was a computing teaching assistant and the Coopers' Company and Coborn School. I assisted with teaching a range of topics, most prevalently Python, to a group of Year 10 students. This essay discusses the teaching I completed, along with the methodologies and resources I used, and an evaluation on how the course went.

## 1 About Coopers' Company and Coborn School

I taught at the Coopers' Company and Coborn School, which is located in Upminster, Essex. I chose this school because I considered it as a potential Sixth Form back in 2014; however I eventually decided against it, due to their weaker Computing department in comparison to the one I chose. This meant that I knew it was somewhere that I could help encourage students into Computer Science.

The school was rated as "Good" by Ofsted in 2017[1], which is the second highest rating (out of four possible). The school has Academy status[2], which means that they get more control over the curriculum they teach. This was great for me, since it allowed me to teach things I thought would be most beneficial to the students, and not have to follow the lessons set out in the national curriculum.

They students at the school range from ages 11-18, since the school also has a Sixth Form, and they are of mixed genders. The academic ratings of the school are pretty good, with 77% of students attaining grade 5 or above in English & Maths GCSEs in the 2018/19 academic year, compared to only 43% on average across England.

# 2 Teaching Summary

## 2.1 Overview

Before my first visit to the school, I had email correspondence with Mr Sheehan, the head of Computing. We had been discussing some potential classes of students I could teach, based on the current needs of the school. He suggested two main possibilities. The first of these was to teach Python to the current class of Year 10 students, who were studying for their GCSEs. The second was to teach JavaScript to the current Year 12 class, who were studying for their A-Levels. He also thought it would be useful for me to help out with Programming Club in addition to this, which takes place on a Tuesday and Thursday lunchtime. After looking at the school's timetabling, as well as my own, the Year 10 class made more sense, since they had a lesson immediately before Programming Club on a Thursday.

We had decided on three areas I would be assisting with at the school, which were:

1. Weekly Year 10 Python lesson on Thursday afternoons
2. Programming Club on Thursday afternoons
3. Lectures about University to Sixth Formers and Form Classes

## 2.2 Year 10 Class (Teaching Project)

As discussed earlier, I was working with a class of Year 10 students who needed to improve their programming skills, particularly in Python. This was in preparation for their Edexcel Computing GCSE examinations. I will go more into detail later as to what my exact teaching project was, however will give an overview here.

### 2.2.1 Reasons for Decision

I had previously discussed with Mr Sheehan what him and the school needed the most help with, and this was suggested. He said that the curriculum didn't cover programming well enough that the students understood it to the extent needed for their exams. For example, the students would have a 1 hour lesson dedicated to 1D arrays, which they found a pretty simple concept and picked up pretty quickly, but then they'd also have the same amount of time scheduled to be taught functions, which are a much conceptually harder topic. We agreed that this was an area which required more specialised teaching, with lessons structured so that the harder topics were taught for longer, to help the students with their understanding in preparation for their examinations. The challenge was to prepare more balanced lessons so that the students had a good understanding of all of the fundamentals of Python.

The current skill level of the students was determined by the limited teaching they had in previous years, however this had been using programs such as Scratch, and so they didn't really have any programming experience before entering their current academic year. They also had the short amount of time they'd had in class to cover the content, however the lessons they'd been taught by Mr Sheehan had left many confused on many of the harder topics, such as 2D arrays, while loops, for loops and functions.

### 2.2.2 Teaching Approach

I decided that the best way to approach teaching would be to use an iterative method. I started off by doing some shadowing. In my first introductory session, I met the teachers I'd be working with, Mr Sheehan and Mrs Foreman, and the students I'd be teaching. In my next session, I shadowed a computing lesson run by Mr Sheehan. I used this to find some good teaching techniques that he used, and some ideas I thought could be engaging and I could reuse. He ran a computing theory lesson, which was different to the programming lessons I would be teaching, but I took note of some techniques he used to help the students learn. One of these was to give the students access to the slides he was using during the lesson whilst he was teaching. This is something commonplace at university, but, at least in my own experience, didn't happen often in schools. However, it allowed students to refer back to the slides when they were doing questions so that they understood the content more thoroughly. This leads on to the other thing I noticed, which was that he taught the lesson for about 15 minutes, and then gave the students a worksheet to complete. However, he just left them to their own devices, so there were some students who weren't really participating, because they were struggling with the questions, and so I knew student interaction would be an area I would be sure to work on, so that they didn't become demotivated.

I then used these ideas to plan my first lesson. I researched the Edexcel GCSE syllabus to find appropriate content, as well as using the ideas Mr Sheehan had suggested. I also wanted to find some other content from their syllabus, which was not explicitly programming, but that I could tie into the lessons, to show some more practical uses. One of the ideas I came up with would be to use encryption, since I could make some fun storylines about hacking which I thought the students might find more engaging.

I then got going on creating the materials for the lessons, and then implementing a lesson. Once I had finished, I'd get some kind of feedback from Mr Sheehan and the students, and use this to work on my next lesson, creating a feedback loop. Examples of this will be discussed in depth in section 3

## 2.3 Programming Club

In addition to my teaching project, I also assisted Mrs Foreman at Programming Club after my lesson with the Year 10s.

### 2.3.1 About Programming Club

Programming Club is open at lunch times on Tuesdays and Thursdays every week. It is for any student in Years 7-11. I attended the Thursday sessions.

The structure of programming club was that it was a place where students could come and work on any programming they were interested in, and get help from me and Mrs Foreman. We would also set challenges they could work on, such as the Matrix 2020 challenge, which was a cyber-security challenge organised by Vodafone.

### 2.3.2 Examples of Teachings

As stated above, students could bring any computing-related work to programming club to work on. Here are some examples of what we worked on:

1. One pair of boys in Year 8 were bought 3D printers for Christmas, just before I started. They attended the first Programming Club I assisted at, and were excited to share some of their creations with me and Mrs Foreman. They said that they planned to sell creations and set up a business based on this. They wanted to create a website, and so I suggested some good resources they could use to do this. They started by using a basic online website creator, but over the weeks migrated this to incorporate their own HTML and CSS code. They also managed to connect the website to a database, and together, in the final few weeks, we set up the website to perform backups of their data to a cloud server, after we talked about the benefits of having backups. Over the weeks, they recruited many more students to join this venture, and eventually had a small business running, where they were selling their creations on their own online e-commerce platform.

2. Another example of a project being worked on was a game created by a Year 10 student call Eleeza. She was also in my Year 10 Python class. She was creating a game that she wanted to enter for the BAFTA Young Game Designers Award. This is "*an initiative by BAFTA that inspires and supports young people to create, develop and present their new game idea to the world*"[3]. Her game was educational in nature, and aimed to teach students the basic Python programming principles.

3. We also ran a few sessions where we would teach some important key computing fundamentals to students that wouldn't usually be taught in school. We did this when there had been discussion on a certain topic in class, and we felt it would be good to shine a light on it. One example was one week when a group of students were talking about "hacking", particularly in reference to a game they were playing. We had a small chat about it that lesson, but to follow up we then did a session where we explained how hacking (in the sense they had brought up) was illegal, and showed a video on this. However, we also explained about white-hat hacking, and had a discussion with the students about this, which they were very interested in. This meant we got to show them some potential practical uses of their programming skills, as well as reinforcing good computing practices.

### 2.3.3 Outcome

There are four main reasons why I think Programming Club was a good place to help out.

1. The first of these was that I got to work with a group of students of different ages to the ones I was teaching in class. This meant that I got to try using different teaching techniques and see what would work best with students of different ages. For example, I found that younger students are much more enthused to learn from more visual activities like games, where they can visualise what their code is doing, such as Scratch or `https://blockly.games`, whereas older and more advanced students learn more from actual coding and creating bigger projects.

2. The next reason was that I could test these different teaching techniques out here, in a place which is optional for the children to attend, and so does not count towards a qualification (such as GCSEs). This meant that if something was not clear, it was not the end of the world, and I could just find another way to convey the information. I could therefore iterate on them and implement the best ones in my lessons.

3. The third reason was that I got to work with a group of students who had a genuine interest in computing, since they were giving up their lunch time to be there.

4. This links to the forth point, which was that I could therefore teach a wide range of different materials, compared to only teaching Python in the Year 10 classes. This meant that I also got to see which styles of teaching worked well for different topics, as well as for different ages, and I also got practice in delivering content that was not programming-focused.

## 2.4 Sixth Form Lectures

The final of my three main teaching areas was giving lectures to Sixth Form students about studying at university, obviously with a specialisation in computing at Imperial.

I gave lectures to the Year 12 and the Year 13 Computing classes, and also the week prior, I got to hold a Q&A with Mrs Foreman's Year 9 form class in order to get an idea of what questions students would have for me.

I gave a talk covering many topics ranging from what I studied, to the admissions process, to how university differs to sixth form, to some generics of university life. However, I found the students benefited most from the Q&A sessions. In these, they asked many questions, ranging from questions on exams, to course options, to societies, to social life. I think that this was an excellent way to demonstrate the differences between how different the teaching of computing are at school and university.

# 3 Teaching Project

I knew I'd be teaching for six weeks, so I startead off by researching a few topics I thought would be good to cover, with the help of Mr Sheehan and the GCSE syllabus. He said that 2D Arrays, for loops, while loops and functions were key weaknesses the class had, so I planned out a lesson structure according to this, the final version of which can be seen in Table 1 below.

| Week | Lesson Title | Topics Covered |
|---|---|---|
| 1 | Arrays | 1D arrays, 2D arrays and iteration on arrays |
| 2 | For Loops vs While Loops | Differences between for loops and while loops |
| 3 | Encryption | Solidifying knowledge on iteration and arrays using Morse code |
| 4 | Functions | Introduction to functions using Caesar Shift Cypher |
| 5 | Further Functions | Solidifying knowledge on functions |
| 6 | Built-in Python Methods | Differences between methods and functions, basic methods |

Table 1: Lesson Plan

I'll now look over what I taught, which materials I used, the feedback I received and the changes I implemented based on this feedback.

## 3.1 Week 1 - Arrays

In my first week, I was teaching about Arrays, with the aim to get the students to particularly learn about 2D arrays, since this had been raised as an area they'd been struggling with. I decided to start off the lesson with a Kahoot, which is an online quiz, to see where all of the students were at. I used this as a baseline to get some feedback on areas they struggled with in relation to arrays, as I knew it was an area they had recently covered in class, and I didn't want to spend lots of pointless time re-teaching material they already knew.

My plan was to give a presentation on the content they needed to cover, and then go around the classroom and help them as they were doing the worksheet I had created for them. However, I found that by the time they got to the questions on the later topics on the worksheet, they'd forgotten what they'd been taught earlier on in the lesson. Another problem was that some students were not moving onto the later topics, since they were spending a long time struggling on the easier questions at the start. I initially tried using Kahoot to get feedback about the lesson, but it wasn't all that useful, and I found having actual conversations with students was a lot more fruitful. What speaking to them about what they thought could have gone better, they reiterated the points I stated above about not getting far enough into the worksheet.

## 3.2 Week 2 - For Loops vs While Loops

Using the feedback from the week before, my plan was to give more broken up presentations on topics, and intersperse these with worksheet questions throughout the lesson. I think that this worked better that the previous week, since the students moved through the worksheet and tried more questions, but I found it a bit tricky to get their attention back for the presentations in the middle, since some of them would just continue working on the previous questions. I also found that a lot of the students had forgotten some of the material from the previous week that we had covered, which was on arrays. A piece of feedback one of the students gave me was that the presentations were a bit long and too detailed, which is why some of them weren't engaged enough, as they went into extra detail that the students didn't need to know.

## 3.3 Week 3 - Encryption

Using the feedback from the previous week, I made the presentations shorter, teaching just the key information they needed. I would then give the more interested students references to places to read more detailed notes if they wanted. One thing I did to try and get the students to remember what they had learnt the week before was to do a short quiz on For Loops vs While Loops, and when each should be used. I did this by making them move around the classroom based on which they thought should be used. This reminded them what we had done the previous week, as this was content they'd previously found difficult, and this seemed to get them engaged and ready to start the lesson with the base knowledge they needed. I decided to use encryption as a mechanism for conveying my lesson plan in this week, where I was trying to solidify the content I had taught in the previous two weeks, but by showing examples of how the content may be used in real life.

However, one thing I was noticing was that some of the students weren't motivated enough to do the worksheets. I spoke to a few of them about this at the end of the lesson, and also asked what they would find fun, and a few brought up that they would be more invested if there were an aspect of competition between the students, and Mr Sheehan agreed with this. I also did some research, which found that competition had been shown to be a good motivator, and had increased learning efficiency[4], however I wanted to make sure not to exclude any of the weaker students by making it intimidating if they could not complete the worksheets.[5]

## 3.4 Week 4 - Functions

Throughout the week before this lesson, I wanted to create a system which would allow for some healthy competition in the lessons. My idea was to find a system which may be similar to HackerRank combined with Kahoot, where they could answer coding questions, and have a leaderboard with the students who have achieved the most correct tests.

I looked for alternatives which already existed, but nothing really fulfilled the requirement I needed, so I created it myself. The process worked using a Jupyter Notebook on Google Colab, which is essentially the programming version of Google Docs. They would be given worksheet questions on this, and would use the blank spaces to write and test their answers. Once they tested their answers, their score was sent to a server, which calculated the leaderboard, and it could be displayed on the screen at the front. It used randomly generated names to ensure that they were not inappropriate, and to encourage the weaker students to contribute and take risks with code, without everyone else knowing how well they were doing. Also, data was only stored temporarily, so once the system hadn't been used for half an hour the data was lost (for GDPR compliance). More technical details can be found in Appendix A.

It took a while for the students to get used to using a new program to code with, but eventually they were using it fluently. I chose to introduce this because I thought it was also good practice for the real world, such as when working in industry, since if they continued Computing after school, they'd most likely be using different IDEs which weren't the default Python shell. However, I did find that with the worksheet, the first few questions were too long, and the majority of the students didn't progress far enough through it, since quite a few of them still didn't have a deep enough grasp of what functions were.

## 3.5 Week 5 - Further Functions

For the following week, I kept with a similar setup as the previous week, but gave the presentation on functions from a more visual perspective, to cater towards the more visual learners in the class. I did this by using diagrams that gave a visualisation of a function taking in data, manipulating it, and giving an output. This made more of the students understand the purpose of functions.

Another change I implemented was to make the tutorial sheet significantly easier for the first few questions, but add a lot more questions that increased in difficulty. This meant that the less able students were able to get started, and didn't lose confidence and motivation early on, but also that the more advanced students didn't finish early and not learn anything, which worked well.

Towards the end of the lesson, me and Mr Sheehan spoke with the students asking which topics they found most difficult throughout the course, to get a good consensus of what would be good to teach in the last session, since I had saved this to cover something of the students' choosing.

## 3.6 Week 6 - Built-in Python Methods

Based on the students' feedback, I taught a lesson on built-in Python methods. I made the lesson follow a similar structure to the previous week, however added a Kahoot quiz at the start in order to reinforce all of the course content which we had covered. I also included some questions on built-in methods, which I had just taught that lesson, since this is quite a memorisation-based topic. The lesson was much more successful than the earlier lessons, with the students completing the majority of the worksheet, and really grasping the concepts well.

# 4 Evaluation

## 4.1 Increase in student understanding

The overarching aim of my teaching project was to improve the knowledge of Python for the Year 10 students. I believe that I achieved this, since by monitoring the difficulty of questions they were answering, and looking at results of Kahoot quizzes, they improved greatly. In week 1, the average percentage of correct answers was 56%, whereas in week 6 it had increased to 66%. This is despite the fact that in the final quiz, there were questions relating to in-built methods, which the students didn't know the answers to. This would have skewed the results down, therefore the fact the result is still higher than week 1 shows an upward trend in Python knowledge.

The full results of the Kahoot quizzes for weeks 1 and 6 can be found in Appendix B.

## 4.2 Common Challenges

There were five main challenges I came across throughout my teaching. These were:

### 4.2.1 Time Management

Time management was something which I initially struggled with. There would be lessons where I'd spend too long on a certain topic, and therefore would not teach everything I had planned for the lesson. However, after a few lessons I found that it was best to not stick to a rigid schedule, and to be flexible with extending parts of the lesson if students need more elaboration on a certain topic, or go quicker through easier topics they already knew, in order for them to not get confused or lose interest. By the end of my course, this meant I had a better understanding of how much time certain things would take, which also came with getting to know the students and their capabilities better.

### 4.2.2 Adapting content to a younger audience

Another challenge I encountered was adapting content that I knew at a university level down to a level that students aged 15 would understand. This ties into point 4.2.5 on catering to different abilities in the class, but it was more general in the fact that I had to check teaching materials with people who didn't know programming, which meant going outside my friends at Imperial and going to younger family members in order to check it was understandable from someone coming with a relatively fresh perspective.

### 4.2.3 Student Engagement

Keeping students engaged was something that took a few weeks to get a grip of. In the first few sessions, I could see some students would lose interest, either because the found the lessons too easy or too difficult. I tried experimenting with how hard I made the worksheets, which helped to an extent, but I think the biggest way I helped to encourage engagement was by introducing an element of competition into the lessons, to make it more interactive and to make them want to participate.

#### 4.2.4 Feedback

Another problem I initially encountered was getting useful feedback from students. I tried using online quizzes, such as Kahoot, but the students didn't really take them seriously, and would give useless answers. Mr Sheehan also seemed to just say everything I was doing was great, since he wasn't an expert in Python programming, so anything I was teaching them seemed good to him. Therefore, I had to find better ways to obtain feedback, such as asking students in person how they found things, in order to improve my lessons. I found usually that they would give a more honest response in person, which was very helpful. Plus, when I got to know the students and their abilities better, I was able to remember who gave good feedback, and figure out where the majority of people were struggling.

#### 4.2.5 Catering to different abilities

The final main challenge I found was gauging how I should cater the lessons to work for the wide variety of skill levels across the class. I initially set my expectations too low, by making the worksheets fairly easy, so they got through them quite quickly. I then adapted this, but ended up making them too difficult, and seemed to over-correct. However over the weeks, by taking on more feedback from students and Mr Sheehan before teaching, I managed to settle on worksheets which best helped to enrich the students' learning.

## 4.3 Final Thoughts & Key Learnings

Something that really surprised me was the technical ability of some of the children at such a young age. Especially in Programming Club, there were students who were creating webpages that used pipelines and databases, things that I had only recently learned at university. This showed their wide range of interests outside of the curriculum, which I think is something important in schools, since these are more of the skills that will differentiate students applying to study Computing at university, rather than just their A Level grades.

I think the ability to teach these students with a passion for the subject was the most rewarding, as they were learning a lot more, since they were genuinely invested. If I were to teach, I think finding things like this, which students are genuinely interested in, is the best way to encourage them to learn, especially combined with using different teaching techniques such as competition, visual teaching, group work and movement. However, this also is the reason why teaching requires a lot of work, since you have to prepare these lessons, and also listen to the feedback to make sure that you're teaching in the best way for the students.

I had a great time teaching. Before this, I was considering it as a potential path further on in my career, and I feel like this has given me a lot more clarity and reassurance. I think that if I were to teach, I'd prefer to work with older children who know they have an interest and want to learn, for example sixth formers or university students, but I think I want to spend time out of academia working in industry first.

## References

[1] Carolyn Dickinson. Short inspection of The Coopers' Company and Coborn School. Ofsted report, OFSTED, May 2017.

[2] Government School Comparison Website. `https://www.compare-school-performance.service.gov.uk/school/136600/the-coopers'-company-and-coborn-school/absence-and-pupil-population`. Accessed: 2020-03-23.

[3] BAFTA Young Game Designers Award. `http://ygd.bafta.org/`. Accessed: 2020-03-24.

[4] Bjarne Skjødt Worm and Steen Vigh Buch. Does competition work as a motivating factor in e-learning? a randomized controlled trial. *PLOS ONE*, 9(1):1–6, 01 2014.

[5] How to motivate students using competitive and collaborative activities. `https://www.lexialearning.com/blog/how-motivate-students-using-competitive-and-collaborative-activities`, January 2020.

# A Online Leaderboard Technical Implementation

The Github Repo for this project can be found at `https://github.com/jackmorrison12/python-leaderboard`.

## A.1 Backend

The online leaderboard was created using a Heroku dyno located at `https://vast-depths-77634.herokuapp.com/`. It used a Python Flask server, which runs on the dyno. Since it is a free dyno, it turns off after 30 minutes of non-usage, which means that any data stored on it is lost after 30 minutes.

I set up endpoints on the flask server to receive POST requests, which can be seen in the app.py file. These received the answers that the students submit, and would return a string with how many answers the student got correct. It would also receive their username, and use a Python dictionary to store their points for each question. This is just stored locally on the Heroku dyno, since the data will only be needed for a short while, and students will be sending POST requests more frequently than every half an hour, so data will be safe for the duration of the lesson, and then wiped after to comply with GDPR.

I did have a problem where there are multiple instances of the dyno running, and therefore data would not be consistent across them all. I found that there would at maxinum be 3 dynos running, and so I sent 20 POST requests from the students, which to my knowledge meant that the multiple dynos were usually consistent. If this system was to be extended, I would connect it to a database in order to reduce this error, however the current setup was sufficient for the lesson being taught.

## A.2 Frontend

The students would complete their work in a Jupyter Notebook using Google Colab. This would have a pre-created worksheet with all of the questions for that session in, which was distributed to students through Google Classroom. They'd make a local copy, and then start filling in the function stubs in order to fulfil each task. Once they had done a question, they could compile the code to check it worked, and then run the test cases for that question. This would send a POST request which contained the username they'd generated, and an array of their answers, to the backend. These would be compared against the sample answers on the server, and the resultant string detailed above would be returned and displayed as the output code.

The leaderboard website displayed the dictionary of names and scores detailed above, in descending order of score. This just works by simply returning the HTML which contained a table populated with the dictionary contents whenever anybody hit the index of the URL.

## B Kahoot Results

| Rank | Player | Total Score (Points) | Correct Answers | Incorrect Answers |
|---|---|---|---|---|
| 1 | Panashe | 4318 | 4 | 0 |
| 2 | Chris S | 4214 | 4 | 0 |
| 3 | Eleeza | 3102 | 3 | 1 |
| 4 | Tom | 3098 | 3 | 1 |
| 5 | David | 2811 | 3 | 1 |
| 6 | Josh | 2010 | 2 | 2 |
| 7 | Hao | 1895 | 2 | 2 |
| 8 | Jeremy | 1892 | 2 | 2 |
| 9 | Frederick | 1789 | 2 | 2 |
| 10 | Ikechi | 948 | 1 | 3 |
| 11 | Joe | 937 | 1 | 3 |
| 12 | Chris S | 0 | 0 | 4 |

Table 2: Table of Kahoot Results for Week 1 - Arrays

| Rank | Player | Total Score (Points) | Correct Answers | Incorrect Answers |
|---|---|---|---|---|
| 1 | Tom | 12517 | 10 | 0 |
| 2 | Chris C | 9639 | 9 | 1 |
| 3 | Frederick | 8065 | 8 | 2 |
| 4 | Ikechi | 7283 | 7 | 3 |
| 5 | Hao | 7123 | 7 | 3 |
| 6 | Jeremy | 6132 | 6 | 4 |
| 7 | Chris S | 6012 | 6 | 4 |
| 8 | David | 4890 | 5 | 5 |
| 9 | Josh | 3599 | 4 | 6 |
| 10 | Joe | 3172 | 4 | 6 |

Table 3: Table of Kahoot Results for Week 6 - In-built Python Methods