which means light is scattered evenly and randomly throughout the liquid. Solids like opals are colloid crystals, which gives the gems their structural color.

These properties can be very useful in modern development. Architects, for example, have designed buildings with iridescent surfaces. However, it is often more useful to have structural color that is angle independent, because, for example, it is easier to read on a surface that does not keep changing color every time you shift your head. Researchers have developed a technique using colloid amorphous arrays to obtain angle independent structural color, but the solution was also background dependent, which means that it would appear white if the background is white but would have color if the background is black. To solve the problem, the researchers have added carbon black to the mixture to enhance the color. A potential use of this color technique is to design building windows that change color based on the environment inside the building to better regulate heating and cooling.

3.5 Friday: Halt!

By Josh M.

Speaker: Emil Guliyev (International Moneymaking Company)

On Friday afternoon, we gathered to hear Emil speak about computers. In fact, he announced this straight away, with a plaintive announcement: "I'm going to talk about computers." This was met with a chuckle from the back of the room, while the students remained in their seats.

Emil's guiding question was, "What is computation?" We explored this by first thinking of things that computers do: E-mail, shopping, Minecraft, and bitwise AND. The ultimate cause of the first three uses is that computers can do three things: make variables, if statements, and while loops. Essentially, all a computer does is start with a program and apply the same rules repeatedly until a result is reached.

To understand an ordinary computer better, we discussed a few theoretical computing systems. The first was known as the "Cellular Automaton" system. One begins with a string of ones and zeros. To obtain the next string, each position is assigned a value based on the three values in the previous string directly above it. He gave us an example of such a rule system:

111	110	101	100	011	010	001	000
0	1	1	0	1	1	1	0

He then moved on to another system, known as the "Two-Tag" system. In this system, one was given an alphabet of letters, including one corresponding to "Halt", a starting string of letters, and production rules. On each step of the computation, the first two letters are removed, and the production rule corresponding to the first letter removed is applied. A production rule adds letters to the end of your string. When the Halt character is reached, or there are zero or one letters left, the computation is over. He used this system to demonstrate that the number eight is a power of two.

He then introduced us to the most common theoretical computing system: the Turing machine. The Turing machine reads an infinite tape split into cells. On each cell is a letter from this alphabet: one, zero, or blank. In addition, the Machine carries its own state. Based on what it

reads and what its state is, the Machine can write a new value, move around, and/or change its state. This system formalizes very easily, but computing with it would take a very long time.

One interesting thing about all of these systems is that they can all do the same computations; they are equally powerful. This is known as being Turing-complete (equivalent to a Turing machine).

Then, we turned to the Halting problem, an old problem in computer science and one of Emil's favorites. The question was, "Given an arbitrary program, can we use a Turing machine to determine whether it will stop?" The answer is no, and Emil gave a proof in Python. Given a halt-checker, he created a program such that when given itself as its own argument, it will only halt if it goes on forever, and it will only continue if it halts. This is a paradox, and so a halt-checker cannot exist.

To spend the last few minutes, we saw a presentation on the history of Computer Science. We saw Leibniz (the creator of binary), Ada Lovelace (considered to be the first computer programmer), Alan Turing (the creator of the Turing machine), and Alonzo Church (his mentor), all huge names in computer science. As a final story, Emil told us of some of the oldest digital computer memory: magnetic rings hand-woven into a wire lattice. We laughed and thought of old computer scientists knitting together their memory for their gargantuan computers, and thought of how far we've come today.

3. Fun Stuff!

4.1 Music that you should check out!

Time to explore new territories of music! Here is a list of songs recommended by your fellow mathletes.

Sarah-marie:	"**Penultimate Persian**" by Chris Clark (good part starts 27 sec in)
Tom:	"**Rainbow Voodoo**" by Chris Clark
	"**Come On My Selector**" by Squarepusher
Reese:	"**Heartbeat City**" by The Cars
Andy:	"**Kodachrome**" by Paul Simon
Emi:	"**The Gambler**" by fun.
Rashmika:	"**Song of the Caged Bird**" by Lindsey Stirling
Ina:	"**Adoremos**"
Cecilia:	"**John and the Monster**" by Ariana Gillis
Gideon:	"**Gun Has No Trigger**" by Dirty Projectors
Sarah:	"**Stop and Stare**" by OneRepublic
Josh:	"**Anything by Beethoven"**
Emil:	"**I ain't about this.**"