Turing Machine

Damian Meza Madrid

September 2018

1 Introduction

In 1900, the International Congress of Mathematicians in Paris, directed by David Hilbert proposed a set of problems for the new year, and the new century. This problems where not mathematical, but *about* mathematics themselves. Here are these questions:

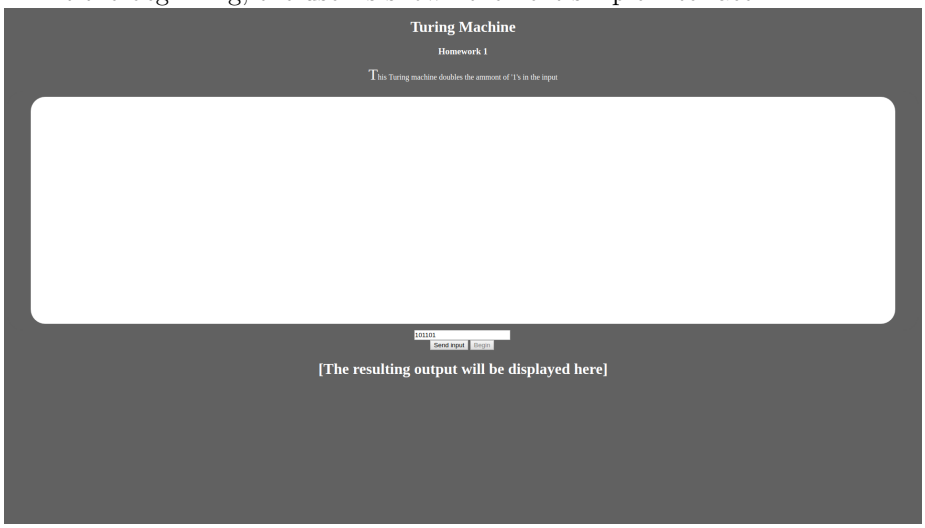
1. Is mathematics complete?
2. Is mathematics consistent?
3. Is every statement in mathematics decidable?

35 years later, Alan Turing, using the answers to the previous questions decided to solve the 3rd problem and in the process developed what would be the blueprint for the invention of the programmable computer, the Turing Machine.[2]

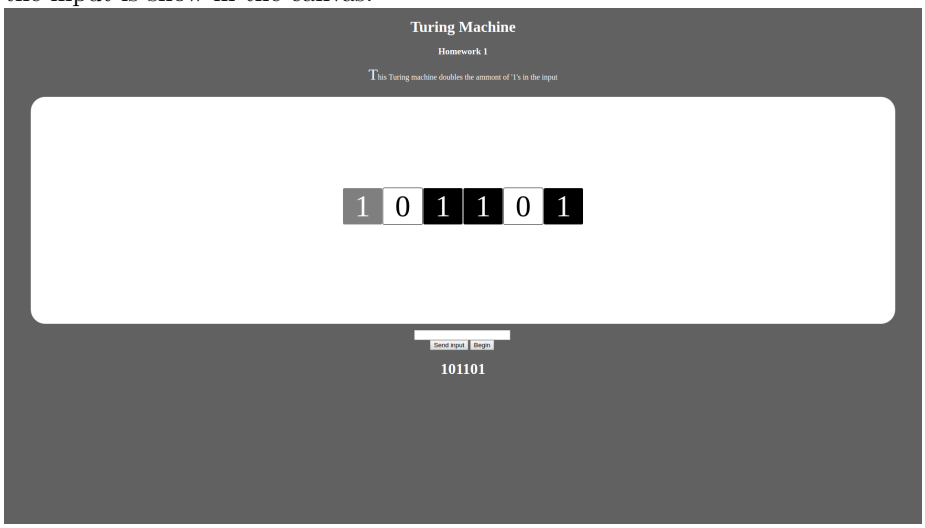
2 The Turing Machine, Implemented in JS

The JavaScript language was used because it's widely know that simple programs that have a UI can be easily written in web technologies as HTML, CSS and JS. At the same time, if necessary, this technologies can be taken to a desktop aplication using the framework Electron.

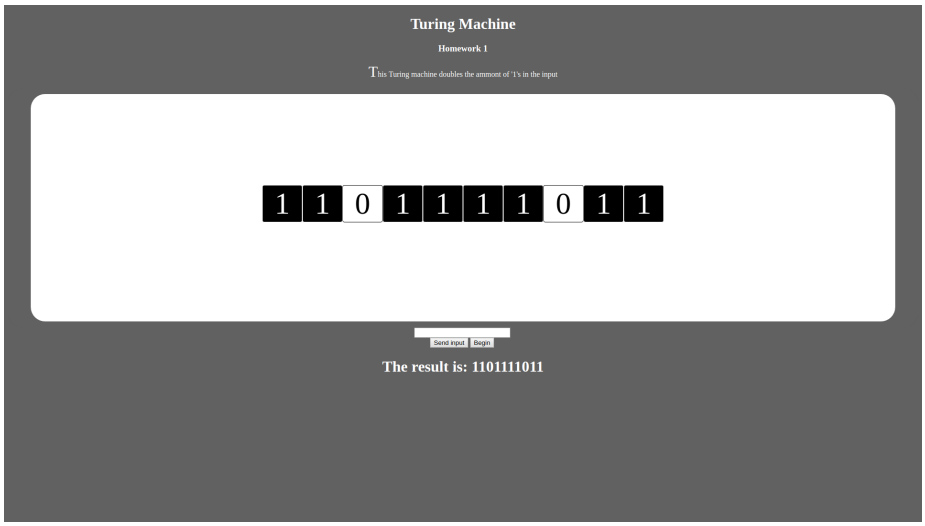
At the beginning, the user is shown the next simple interface:



The user then types the input in the textbox, and presses Send Input, so that the input is show in the canvas.



Afther the input has been submited, the user will press Begin, the animation will start and the resulting input will be displayed in the bottom along with the tape



3 Source Code

All the source Code for this and other projects for this class can be found on: https://github.com/asdf1234Damian/ComputerSelectedTopics.

3.1 HTML Objects

This implementation of the Turing machine uses the class el, for element, as the basic cell of the tape. Each element is followed by the kind of symbol it has, white for 0 and black for one. grey is used to identify the current symbol being read by the machine :

```
<div class="el grey">"+currentTape[i]+"</div>
<div class="el black">"+currentTape[i]+"</div>
<div class="el white">"+currentTape[i]+"</div>
```

Together, the element, form the tape, which are handled by JavaScript togheter with the anime.js library[1]

3.2 Javascript Code

Next is the whole javascript file for the program, the HTML and CSS files are located in the resource https://github.com/asdf1234Damian/ComputerSelectedTopics.

```
var currentTape;
var currentPos = 0;
var currentState;
var stack;
```

```
var printing=false;
var running=false;

function displayTape() {
  console.log("Displaying("+currentTape.join("")+","+currentPos+")");
  document.getElementById("mid").innerHTML= "";
  for (var i = 0; i<currentTape.length; i++) {
    if (i==currentPos) {
      document.getElementById("mid").innerHTML+= "<div class=\"el grey\">"+currentTape[i]+"<
    }else{
      if (currentTape[i]=="1") {
        document.getElementById("mid").innerHTML+= "<div class=\"el black\">"+currentTape[i]
      }else{
        document.getElementById("mid").innerHTML+= "<div class=\"el white\">"+currentTape[i]
      }
    }
  }
  document.getElementById("result").innerHTML=currentTape.join("");
}


function addTape() {
  console.log("Moving input to tape");
  currentTape=document.getElementById('inputtape').value.split("");
  currentPos=0;
  displayTape();
  document.getElementById("turing").disabled=false;
  document.getElementById("inputtape").value="";
}

function moveRight() {
  console.log("MovingRight");
  anime({
    targets: '.el',
    translateX:{
      value:'-=80',
      duration:1500
    }, complete (anim){

    }
  });
  currentPos =currentPos+ 1;
  displayTape();
}
```

```
function stopMachine() {
  console.log("The Turing Machine stopped");
  document.getElementById('inputtape').disabled=false;
  document.getElementById('turing').disabled=false;
  document.getElementById('sendbtn').disabled=false;
  document.getElementById("result").innerHTML="The result is: "+currentTape.join("");
}

function Turing(){
  if (currentPos >=currentTape.length) {
      for (var i = 0; i < stack.length; i++) {
        currentTape.splice(currentPos,0,1);
        stack.pop();
        moveRight();
      }
      //Automatic Start
      //displayTape();
      //return stopMachine();
  }
  switch (currentState) {
    case 0://Start
    if (currentTape[currentPos]=='0') {
      moveRight();
      Turing();
    }else{
      stack.push(1);
      currentState=1;
      console.log("stack:"+stack);
      moveRight();
      Turing();
    }
    break;

    case 1:
    if (currentTape[currentPos]=='0') {
      for (var i = 0; i < stack.length; i++) {
        currentTape.splice(currentPos,0,1);
        stack.pop();
        currentPos++;
      }
      stack=["z"]
      currentState=0;
      moveRight();
      Turing();
    }else{
      stack.push(1);
```

```
      console.log("stack:"+stack);
      moveRight();
      Turing();
    }
    break;
  }
}

function startMachine() {
  document.getElementById('inputtape').disabled=true;
  document.getElementById('turing').disabled=true;
  document.getElementById('sendbtn').disabled=true;
  stack=["z"]
  currentState=0;
  Turing();
}
```

4 Conclusion

The Turing Machine is easy to implement, not so easy to animate, in modern programming languages, but works as a good, introductory project for the next project, Cellular Automaton, both of them capable of universal computation [2]

References

[1] http://animejs.com/.

[2] Melanie Mitchell. *Complexity: A Guided Tour*. Oxford University Press, 2011.