Politechnika Śląska

Gliwice, 2019-01-24

Principles and applications of mathematical modeling

The Monte Carlo Method

Mateusz Dobija

I. For users

a) *Description of the program*

The Monte Carlo method helps us to calculate the approximate value of π. The logic of the program depends on the randomness of the numbers from -1 to 1, then check whether the drawn value belongs to the circle or whether it is outside. The π result depends on: points inside the circle and all points (inside and outside).

Formula: $\boldsymbol{\pi = \frac{insidePoints}{totalPoints} * 4.}$

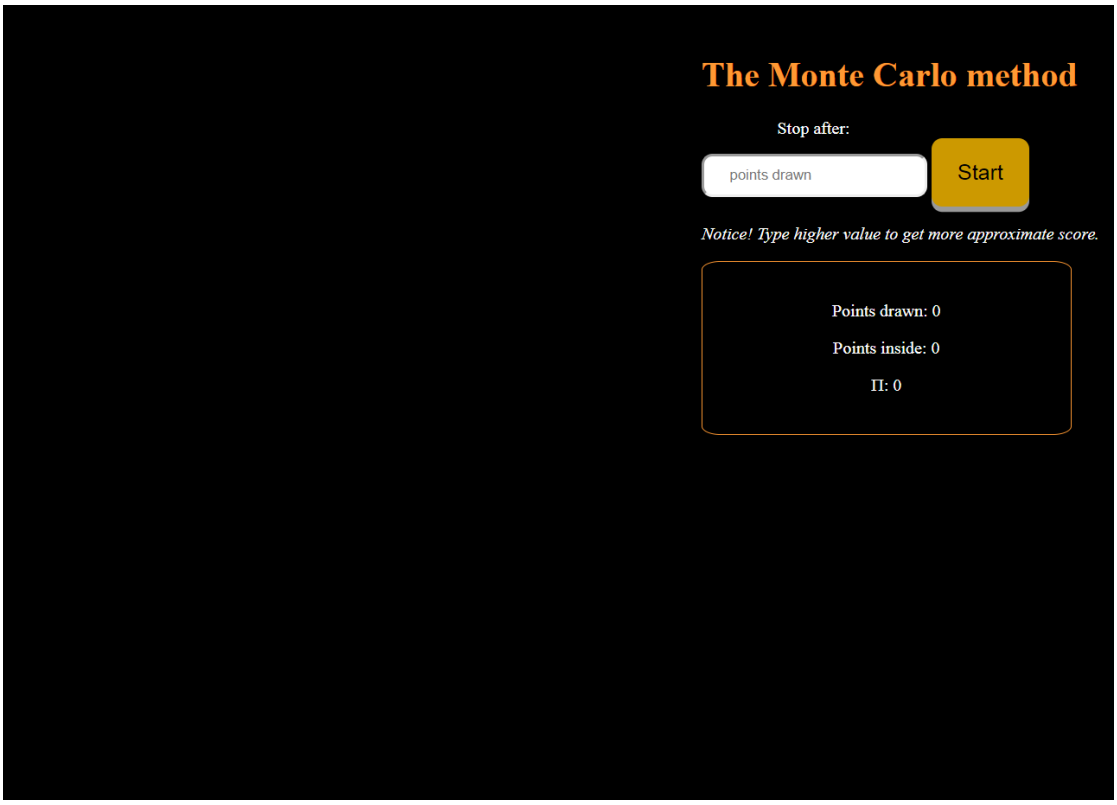
The program is multi-platform, it can be installed on:

- Linux,
- Windows,
- IOS.

b) *Operating manual*

The service is very simple. You have to run the file with the .html extension in your browser. Then type the value of points drawn and click START. Program will be run! On the left side, a circle on a square background will begin to draw. In addition, we see the number of all points, the number of points inside the circle and the approximate value of Pi. For a more accurate result, type the largest possible value in the "stop after" field.

View of the program that has not been started

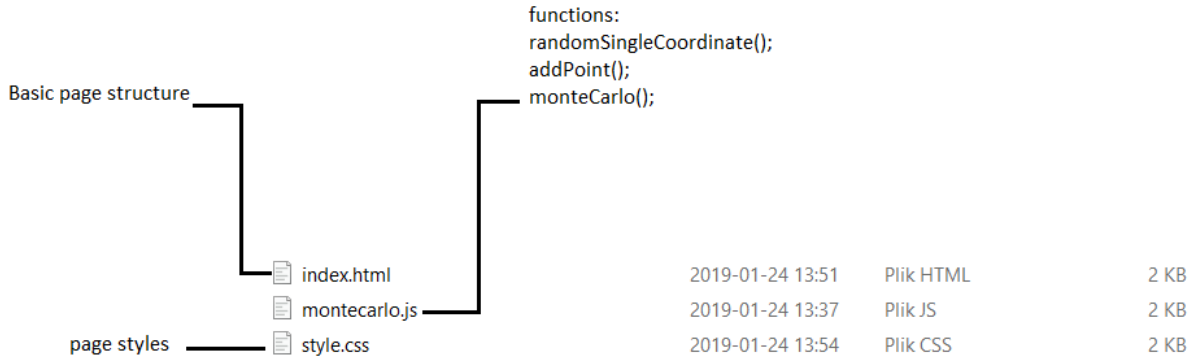


View of the running program

II. For programmers

a) Technical specifications

- Division for the files:



- Used technologies:

- Programming environment:

Visual Studio Code

b) Technical details

- Main functions:

```
function randomSingleCoordinate(){
    return((Math.random() * 2) - 1);
}
```

In this function, we randomly draw any point from -1 to 1.

```
function addPoint(){
    let x = randomSingleCoordinate(), y = randomSingleCoordinate(),
        is_inside_circle = (((x*x)+(y*y)) < 1);

    total++;
    if(is_inside_circle) inside++;
    if(show_graphics){
        let point = document.createElementNS("http://www.w3.org/2000/svg", 'rect');
        point.setAttribute('x', (x + 1) * 300);
        point.setAttribute('y', (y + 1) * 300);
        point.setAttribute('width', 1);
        point.setAttribute('height', 1);
        if(is_inside_circle) point.setAttribute('class', 'inside');
        svg.appendChild(point);
  }
}
```

In this function, we check whether the point belongs to the circle and we draw.

```
function monteCarlo(){
  var pointsToAdd = 1;
  if(total > 0 && perform_acceleration) pointsToAdd = Math.pow((Math.floor(Math.log10(total))+1),3);
  // add one or more points
  for(var i = 0; i < pointsToAdd; i++) addPoint();
  // Update output
  output_total.innerHTML = total;
  output_inside.innerHTML = inside;
  output_pi.innerHTML = inside / total * 4;
  // Loop
  let stop_after = document.querySelector("#stopAfter").value;
  if(total < stop_after) setTimeout(monteCarlo, 0.5);
}
```

The main function in which we count the results, we put on the screen, set the duration of the function.