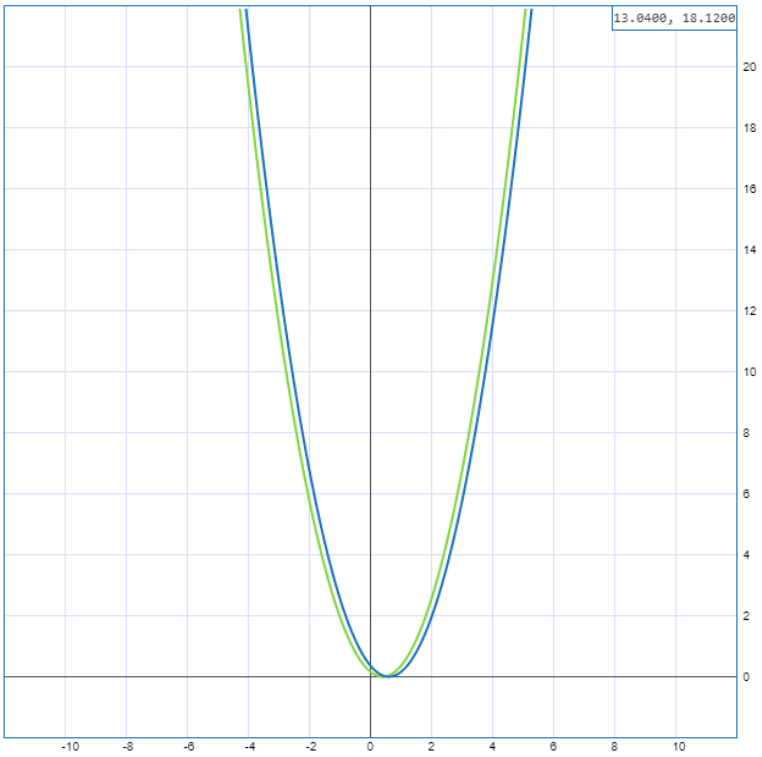
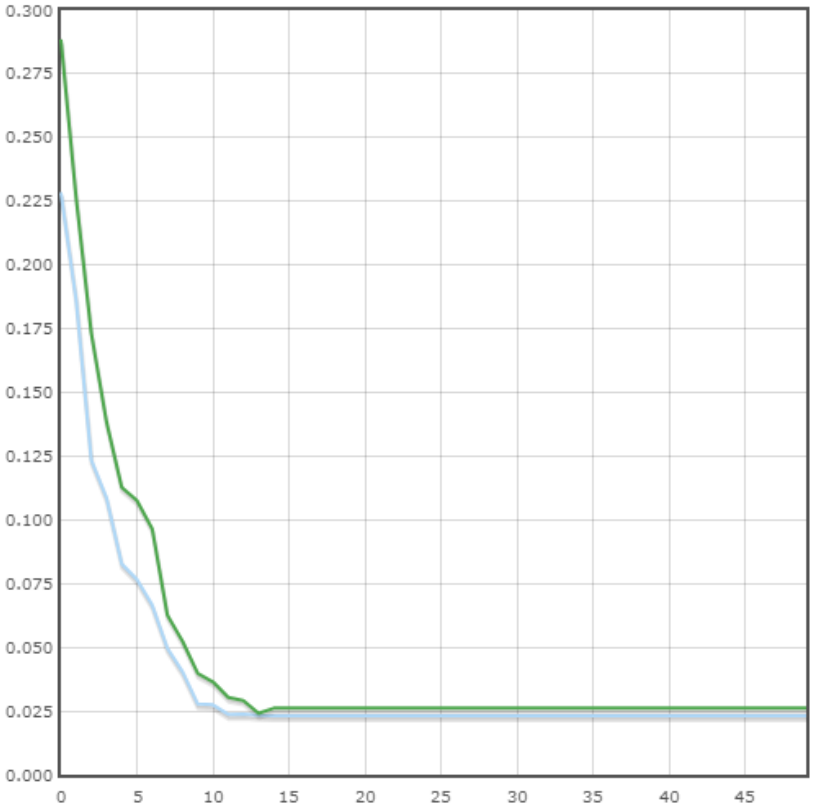
GREEN LINE - (x-0.4)^2
BLUE LINE - (x-0.6)^2



GRAPHICS



VALUE:

0 : 0.22847932	13 : 0.0232465	26 : 0.023051862	39 : 0.023051862
1 : 0.18643243	14 : 0.023100019	27 : 0.023051862	40 : 0.023051862
2 : 0.12344154	15 : 0.023058485	28 : 0.023051862	41 : 0.023051862
3 : 0.10857838	16 : 0.023051862	29 : 0.023051862	42 : 0.023051862
4 : 0.08274684	17 : 0.023051862	30 : 0.023051862	43 : 0.023051862
5 : 0.07661946	18 : 0.023051862	31 : 0.023051862	44 : 0.023051862
6 : 0.0665834	19 : 0.023051862	32 : 0.023051862	45 : 0.023051862
7 : 0.04961762	20 : 0.023051862	33 : 0.023051862	46 : 0.023051862
8 : 0.04037243	21 : 0.023051862	34 : 0.023051862	47 : 0.023051862
9 : 0.027727925	22 : 0.023051862	35 : 0.023051862	48 : 0.023051862
10 : 0.027442123	23 : 0.023051862	36 : 0.023051862	49 : 0.023051862
11 : 0.02363644	24 : 0.023051862	37 : 0.023051862	
12 : 0.02371952	25 : 0.023051862	38 : 0.023051862	

CODE:

```
var Population = function(params) {
    this.populationArray = [];
    this.newPopulation = [];
    this.size = params.size;
    this.params = params;
};

Population.prototype.init = function() {
    for (var i = 0; i < this.size; i++) {
        var tmp = new Individual();
        tmp.init(this.params);
        tmp.fitSum();
        tmp.toBinary(this.params);
        this.populationArray.push(tmp);
    }
};

Population.prototype.sort = function() {
    this.newPopulation = this.newPopulation.sort(function(a, b) {
        return b.bigFit - a.bigFit;
    });
    this.populationArray = this.populationArray.sort(function(a, b) {
        return b.bigFit - a.bigFit;
    });
};

Population.prototype.selection = function() {
    this.populationArray = this.newPopulation.slice(0, this.size);
};

Population.prototype.crossover = function() {
    this.newPopulation = this.populationArray.slice(0, this.size);
    while (this.newPopulation.length < this.size*2) {
```

```
      var   parentIndex1 = random(0, this.size-1, 0),
            parentIndex2 = random(0, this.size-1, 0),
            parent1 = this.populationArray[parentIndex1],
            parent2 = this.populationArray[parentIndex2],
            child1 = new Individual(),
            child2 = new Individual();

            for (var j = 0; j < parent1.decimals.length; j ++) {
               var pivot = random(1, parent1.binaries[j].length, 0);
               child1.binaries[j] = parent1.binaries[j].substr(0, pivot) +
parent2.binaries[j].substr(pivot, parent1.binaries[j].length+1-pivot);
               child2.binaries[j] = parent2.binaries[j].substr(0, pivot) +
parent1.binaries[j].substr(pivot, parent1.binaries[j].length+1-pivot);
           }

           child1.toDecimal(this.params);
           child2.toDecimal(this.params);
           child1.fitSum();
           child2.fitSum();

           if (child1.getSum() <= this.params.budget){
             this.newPopulation.push(child1);
          }
          if (child2.getSum() <= this.params.budget){
            this.newPopulation.push(child2);
         }

   }
};
```