Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение высшего образования
«Московский государственный технический университет имени Н.Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ «Информатика и системы управления»
КАФЕДРА «Программное обеспечение ЭВМ и информационные технологии»

ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №1

По курсу: "Архитектура ЭВМ"

Студент Наместник Анастасия
Группа ИУ7-53Б
Название предприятия МГТУ им. Н. Э. Баумана, каф. ИУ7
Тема Создание хранилищ. Создание классов. Таймер.

Студент: _____ Наместник А.А.
подпись, дата Фамилия, И.О.
Преподаватель: _____ Попов А. Ю.
подпись, дата Фамилия, И. О.

Москва — 2020 г.

TASK_1. Задание 1

Техническое задание:

— Создать хранилище в оперативной памяти для хранения информации о детях.

— Необходимо хранить информацию о ребенке: фамилия и возраст.

— Необходимо обеспечить уникальность фамилий детей.

Реализовать функции:

— CREATE READ UPDATE DELETE для детей в хранилище

— Получение среднего возраста детей

— Получение информации о самом старшем ребенке

— Получение информации о детях, возраст которых входит в заданный отрезок

— Получение информации о детях, фамилия которых начинается с заданной буквы

— Получение информации о детях, фамилия которых длиннее заданного количества символов

— Получение информации о детях, фамилия которых начинается с гласной буквы

Листинг 1 — Код программы. Задание 1.

```
1   "use strict";
2
3   class Children {
4       constructor() {
5           this.array = [];
6       }
7       Add(lastname, age) {
8           if (!(this.array.find(x => x.lastname === lastname))) {
9               this.array.push({ lastname, age });
10          }
11      }
12      Read(lastname) {
13          return (this.array.find(x => x.lastname === lastname));
14      }
15      Update(lastname, new_age) {
16          let child = this.Read(lastname);
17          if (child) {
18              child.age = new_age;
19          }
20          else {
21              console.log("Wrong lastname! (child was not found)\n");
22          }
```

```
23          }
24          Delete(lastname) {
25              this.array = this.array.filter(x => x.lastname !== lastname);
26          }
27          PrintAll() {
28              console.log("\n");
29              for (let i = 0; i < this.array.length; i++)
30                  console.log("Lastname: " + this.array[i].lastname + " Age:
                        " + this.array[i].age);
31          }
32          PrintChild(child) {
33              console.log("Lastname: " + child.lastname + " Age: " +
                    child.age);
34          }
35          GetAverage() {
36              let avg = 0;
37              let len = this.array.length;
38              for (let i = 0; i < len; i++) avg += this.array[i].age;
39              avg /= len;
40
41              console.log("\nAverage age is " + avg);
42          }
43
44          GetOldestChild() {
45              let max = 0;
46              let index = 0;
47              for (let i = 0; i < this.array.length; i++)
48                  if (this.array[i].age > max) {
49                      max = this.array[i].age;
50                      index = i;
51                  }
52              console.log("\nThe oldest child is ") +
                    this.PrintChild(this.array[index]);
53          }
54          GetChildAgeIntervel(x, y) {
55
56              let flag = false;
57              console.log("\n");
58              for (let i = 0; i < this.array.length; i++) {
59                  if (this.array[i].age >= x && this.array[i].age <= y) {
60                      flag = true;
61                      console.log("Child at age within the interval [" + x +
                            ";" + y + "] is");
62                      this.PrintChild(this.array[i]);
63                  }
64              }
65              if (!(flag)) {
```

```
66                 console.log("There are no children at age within the
                       interval [" + x + ";" + y + "]!");
67             }
68         }
69         GetChildByLetter(letter) {
70
71             let flag = false;
72             console.log("\n");
73             for (let i = 0; i < this.array.length; i++)
74                 if (this.array[i].lastname[0].toLowerCase() ===
                       letter.toLowerCase()) {
75                     flag = true;
76                     console.log("Child whose lastname starts with letter "
                           + letter + " is");
77                     this.PrintChild(this.array[i]);
78                 }
79
80             if (!(flag)) {
81                 console.log("\nThere are no children whose lastname starts
                       with letter " + letter);
82             }
83         }
84         GetChildByLastnameLength(x) {
85
86             let flag = false;
87             console.log("\n");
88             for (let i = 0; i < this.array.length; i++)
89                 if (this.array[i].lastname.length > x) {
90                     flag = true;
91                     console.log("Child whose lastname is more than " + x +
                           " is");
92                     this.PrintChild(this.array[i]);
93                 }
94
95             if (!(flag)) {
96                 console.log("\nThere are no children whose lastname is
                       more than " + x);
97             }
98         }
99         GetChildVowel() {
100            function isVowel(letter) {
101                return ['a', 'e', 'i', 'o',
                       'u'].indexOf(letter.toLowerCase()) !== -1
102            }
103
104            let flag = false;
105            console.log("\n");
```

```
106             for (let i = 0; i < this.array.length; i++)
107                 if (isVowel(this.array[i].lastname[0])) {
108                     flag = true;
109                     console.log("Child whose lastname starts with a vowel
                            letter is");
110                     this.PrintChild(this.array[i]);
111                 }
112             if (!(flag)) {
113                 console.log("\nThere are no children whose lastname starts
                        with a vowel letter");
114             }
115         }
116     }
```

Листинг 2 — Код тестов. Задание 2.

```
1  let C = new Children();
2  C.Add("Kim", 12);
3  C.Add("Ripson", 13);
4  C.Add("Bronx", 15);
5  C.Add("Grace", 12);
6  C.Add("Kim", 14);
7  C.Add("Owen", 10);
8  C.Add("Uris", 16);
9  C.PrintAll()
10
11 C.Delete("Bronx");
12 C.PrintAll()
13 C.GetAverage()
14 C.GetOldestChild()
15 C.GetChildAgeIntervel(5, 10);
16 C.GetChildByLetter("g");
17 C.GetChildByLastnameLength(4);
18 C.GetChildVowel();
```

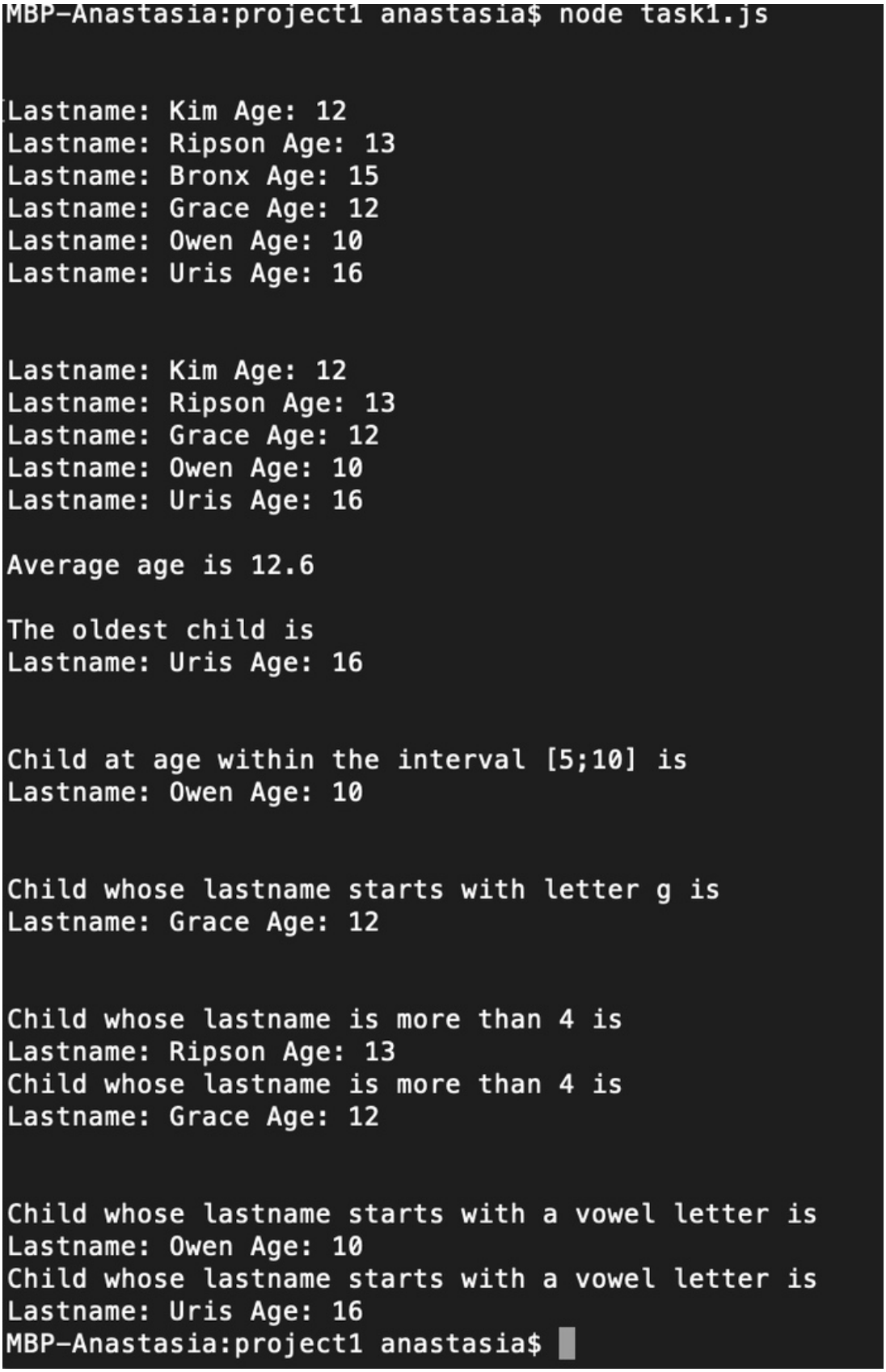

Рисунок 0.1 — Пример работы программы

TASK_1. Задание 3

Техническое задание:

— Создать хранилище в оперативной памяти для хранения точек.

— Неоходимо хранить информацию о точке: имя точки, позиция X и позиция Y.

— Необходимо обеспечить уникальность имен точек.

Реализовать функции:

— CREATE READ UPDATE DELETE для точек в хранилище

— Получение двух точек, между которыми наибольшее расстояние

— Получение точек, находящихся от заданной точки на расстоянии, не превышающем заданную константу

— Получение точек, находящихся выше / ниже / правее / левее заданной оси координат

— Получение точек, входящих внутрь заданной прямоугольной зоны

Листинг 3 — Код программы. Задание 3

```
1   "use strict";
2
3   class Points {
4       constructor() {
5           this.array = [];
6       }
7       Add(name, x, y) {
8           if (!(this.array.find(x => x.name === name))) {
9               this.array.push({ name, x, y });
10          }
11      }
12      Read(name) {
13          return (this.array.find(x => x.name === name));
14      }
15      Update(name, new_x, new_y) {
16          let p = this.Read(tname);
17          if (p) {
18              p.x = new_x;
19              p.y = new_y;
20          }
21          else {
22              console.log("Wrong name! (point was not found)\n");
23          }
24      }
25      Delete(name) {
```

```
26              this.array = this.array.filter(x => x.name !== name);
27          }
28          PrintAll() {
29              console.log("\n");
30              for (let i = 0; i < this.array.length; i++)
31                  console.log("Name: " + this.array[i].name + " X: " +
                        this.array[i].x + " Y: " + this.array[i].y);
32          }
33          PrintPoint(p) {
34              console.log("Name: " + p.name + " X: " + p.x + " Y: " + p.y);
35          }
36          GetPointsMaxDistance() {
37              let max = 0;
38              let index1 = 0, index2 = 0;
39              let len = this.array.length;
40              for (let i = 0; i < len - 1; i++)
41                  for (let j = i + 1; j < len; j++) {
42                      let a = this.array[i].x - this.array[j].x;
43                      let b = this.array[i].y - this.array[j].y;
44                      let dist = Math.sqrt(a * a + b * b);
45                      if (dist > max) {
46                          max = dist;
47                          index1 = i;
48                          index2 = j;
49                      }
50                  }
51              console.log("\nPoints with the maximum distance between them
                    are ");
52              this.PrintPoint(this.array[index1]);
53              this.PrintPoint(this.array[index2]);
54          }
55
56          ConstDistance(x, y, const_dist) {
57
58              let flag = false;
59              console.log("\n");
60              for (let i = 0; i < this.array.length; i++) {
61                  let a = this.array[i].x - x;
62                  let b = this.array[i].y - y;
63                  let dist = Math.sqrt(a * a + b * b);
64                  if (dist <= const_dist) {
65                      flag = true;
66                      console.log("Point which distance to point [" + x +
                            ";" + y + "] is no more than " + const_dist + " is
                            ");
67                      this.PrintPoint(this.array[i]);
68                  }
```

```
69              }
70              if (!(flag)) {
71                  console.log("\nThere are no points which distance to point
                        [" + x + ";" + y + "] is no more than " + const_dist);
72              }
73          }
74
75          GetPointsAxis(axis) {
76              let func;
77
78              if (!axis) {
79                  console.log("\nX axis");
80                  console.log("Above:");
81                  for (let i = 0; i < this.array.length; i++)
82                      if (this.array[i].y > 0)
83                          this.PrintPoint(this.array[i]);
84                  console.log("Below:");
85                  for (let i = 0; i < this.array.length; i++)
86                      if (this.array[i].y < 0)
87                          this.PrintPoint(this.array[i]);
88              }
89              else {
90                  console.log("\nY axis");
91                  console.log("Left:");
92                  for (let i = 0; i < this.array.length; i++)
93                      if (this.array[i].x < 0)
94                          this.PrintPoint(this.array[i]);
95                  console.log("Right:");
96                  for (let i = 0; i < this.array.length; i++)
97                      if (this.array[i].x > 0)
98                          this.PrintPoint(this.array[i]);
99              }
100         }
101         GetPointsInsideRectangle(min_x, max_x, min_y, max_y) {
102             let arr = this.array.filter(p =>
103                 p.x > min_x && p.x < max_x &&
104                 p.y > min_y && p.y < max_y);
105
106             console.log("\nRectangle: -5, 5, -5, 5");
107             for (let i = 0; i < arr.length; i++)
108                 this.PrintPoint(arr[i]);
109         }
110
111     }
```

Листинг 4 — Код тестов. Задание 3.

```
let P = new Points();
P.Add("a", 0, 0);
P.Add("b", 1, 5);
P.Add("c", 2, 4);
P.Add("d", 10, 7);
P.Add("e", 1, -1);
P.Add("f", 0, 3);
P.Add("g", 8, -5.5);
P.PrintAll()

P.Delete("f");
P.PrintAll()

P.GetPointsMaxDistance();
P.ConstDistance(0, 0, 5);
P.GetPointsAxis(0);
P.GetPointsAxis(1);
P.GetPointsInsideRectangle(-5, 5, -5, 5);
```

```
Name: a X: 0 Y: 0
Name: b X: 1 Y: 5
Name: c X: 2 Y: 4
Name: d X: 10 Y: 7
Name: e X: 1 Y: -1
Name: f X: 0 Y: 3
Name: g X: 8 Y: -5.5


Name: a X: 0 Y: 0
Name: b X: 1 Y: 5
Name: c X: 2 Y: 4
Name: d X: 10 Y: 7
Name: e X: 1 Y: -1
Name: g X: 8 Y: -5.5

Points with the maximum distance between them are
Name: d X: 10 Y: 7
Name: g X: 8 Y: -5.5


Point which distance to point [0;0] is no more than 5 is
Name: a X: 0 Y: 0
Point which distance to point [0;0] is no more than 5 is
Name: c X: 2 Y: 4
Point which distance to point [0;0] is no more than 5 is
Name: e X: 1 Y: -1

X axis
Above:
Name: b X: 1 Y: 5
Name: c X: 2 Y: 4
Name: d X: 10 Y: 7
Below:
Name: e X: 1 Y: -1
Name: g X: 8 Y: -5.5

Y axis
Left:
Right:
Name: b X: 1 Y: 5
Name: c X: 2 Y: 4
Name: d X: 10 Y: 7
Name: e X: 1 Y: -1
Name: g X: 8 Y: -5.5

Rectangle: -5, 5, -5, 5
Name: a X: 0 Y: 0
Name: c X: 2 Y: 4
Name: e X: 1 Y: -1
MBP-Anastasia:project1 anastasia$
```

Рисунок 0.1 — Пример работы программы

TASK_2. Задание 1

Техническое задание:

— Создать класс Точка.

— Добавить классу точка Точка метод инициализации полей и метод вывода полей на экран

— Создать класс Отрезок.

— У класса Отрезок должны быть поля, являющиеся экземплярами класса Точка.

— Добавить классу Отрезок метод инициализации полей, метод вывода информации о полях на экран, а так же метод получения длины отрезка.

Листинг 5 — Код программы. Задание 1.

```
1   class Point {
2       constructor(x, y) {
3           this.set_data(x, y);
4       }
5
6       set_data(x, y) {
7           this.x = x;
8           this.y = y;
9       }
10
11      log() {
12          console.log("X: ", this.x, "Y: ", this.y);
13      }
14  }
15
16  class Line {
17      constructor(x1, y1, x2, y2) {
18          this.set_data(x1, y1, x2, y2);
19      }
20
21      set_data(x1, y1, x2, y2) {
22          this.start_point = new Point(x1, y1);
23          this.end_point = new Point(x2, y2);
24      }
25
26      get_distance() {
27          const dx = this.start_point.x - this.end_point.x;
28          const dy = this.start_point.y - this.end_point.y;
29          return Math.sqrt(dx * dx + dy * dy);
30      }
31
```

```
32          log() {
33              console.log("Начальная точка:");
34              this.start_point.log();
35              console.log("Конечная точка:");
36              this.end_point.log();
37          }
38      }
```

TASK_2. Задание 2

Техническое задание:

— Создать класс Треугольник.

— Класс Треугольник должен иметь поля, хранящие длины сторон треугольника.

Реализовать следующие методы:

— Метод инициализации полей

— Метод проверки возможности существования треугольника с такими сторонами

— Метод получения периметра треугольника

— Метод получения площади треугольника

— Метод для проверки факта: является ли треугольник прямоугольным

Листинг 6 — Код программы. Задание 1.

```
    //Task #2.1
"use strict";

class Point {

    constructor(x, y, z) {
        this.init(x, y, z);
    }

    init(x, y, z) {
        this.x = x;
        this.y = y;
        this.z = z;
    }
    renderFields() {
        let messageX = "coordinate X = " + this.x;
        let messageY = "coordinate Y = " + this.y;
        let messageZ = "coordinate Z = " + this.z;
        let fullMessage = messageX + " " + messageY + " " + messageZ;
        console.log(fullMessage);
    }
}

class Offcut {
    constructor(x1, y1, z1, x2, y2, z2) {
        this.x1 = x1;
        this.y1 = y1;
        this.z1 = z1;
```

```
29          this.x2 = x2;
30          this.y2 = y2;
31          this.z2 = z2;
32          this.firstPoint = new Point();
33          this.secondPoint = new Point();
34      }
35
36      getSquareDiff(a, b) {
37          return Math.pow((a - b), 2);
38      }
39
40      getSum(a, b, c) {
41          return a + b + c;
42      }
43
44      renderFields(a) {
45          let message = "Length of the offcut is " + a;
46          console.log(message);
47      }
48
49      getLength() {
50          this.firstPoint.init(this.x1, this.y1, this.z1);
51          this.secondPoint.init(this.x2, this.y2, this.z2);
52
53          let l =
                Math.sqrt(this.getSum(this.getSquareDiff(this.firstPoint.x,
                this.secondPoint.x),
54               this.getSquareDiff(this.firstPoint.y, this.secondPoint.y),
55               this.getSquareDiff(this.firstPoint.z, this.secondPoint.z)));
56
57          this.renderFields(l);
58
59      }
60  }
61
62  //Task #2
63  class Triangle {
64      constructor(a, b, c) {          //1: Initialization
65          this.a = a;
66          this.b = b;
67          this.c = c;
68      }
69
70      checkExist() {
71          let message = "This triangle does not exist";
72          if (this.a + this.b > this.c && this.b + this.c > this.a && this.a
                + this.c > this.b) {
```

```
73              message = "This triangle exists";
74          }
75          console.log(message);
76          //2: Check if exists
77      }
78
79      getPerimeter() {
80          let p = this.a + this.b + this.c;
81          return p;
82          //3: Perimeter
83      }
84
85      getSquare() {
86          let p = this.getPerimeter() / 2;
87          let S = Math.sqrt(p * (p - this.a) * (p - this.b) * (p - this.c));
88          console.log(S);
89          //4: Square
90      }
91
92      checkRightTriangle() {
93          let message = "This triangle is not right";
94          const eps = 1e-2;
95
96          if (Math.abs(this.a * this.a + this.b * this.b - this.c * this.c)
                < eps ||
97              Math.abs(this.b * this.b + this.c * this.c - this.a * this.a)
                    < eps ||
98              Math.abs(this.a * this.a + this.c * this.c - this.b * this.b)
                    < eps) {
99              message = "This triangle is right";
100         }
101         console.log(message);
102         //5: Check if right
103     }
104 }
```

TASK_2. Задание 3

Техническое задание:

— Реализовать программу, в которой происходят следующие действия:
— Происходит вывод целых чисел от 1 до 10 с задержками в 2 секунды.
— После этого происходит вывод от 11 до 20 с задержками в 1 секунду.
— Потом опять происходит вывод чисел от 1 до 10 с задержками в 2 секунды.
— После этого происходит вывод от 11 до 20 с задержками в 1 секунду.
— Это должно происходить циклически.

Листинг 7 — Код программы. Задание 3

```
    //Task #2.2

"use strict";

function firstSet() {

    let n = 0;
    let interval = setInterval(() => {
        n++;
        let message = "number: " + n;
        console.log(message);
        if (n === 10) {
            clearInterval(interval);
            secondSet();
        }
    }, 500);
}

function secondSet() {

    let n = 10;
    let interval = setInterval(() => {
        n++;
        let message = "number: " + n;
        console.log(message);
        if (n === 20) {
            clearInterval(interval);
            firstSet();
        }
    }, 1000);
}
```