練習課題 1

次のそれぞれの関数の計算量がどの程度になるか予想して、O 記法で表してみてください。

関数 1

```
1  function uniq1(array) {
2    const knownElements = {};
3    const uniquedArray = [];
4    for (const elem of array) {
5      if (elem in knownElements)
6        continue;
7      uniquedArray.push(elem);
8      knownElements[elem] = true;
9    }
10   return uniquedArray;
11 }
```

関数 2

```
1  function uniq2(array) {
2    const uniquedArray = [];
3    for (const elem of array) {
4      if (uniquedArray.indexOf(elem) < 0)
5        uniquedArray.push(elem);
6    }
7    return uniquedArray;
8  }
```

関数 3

```
1  function uniq3(array) {
2    const knownElements = new Set();
3    for (const elem of array) {
4      knownElements.add(elem);
5    }
6    return Array.from(knownElements);
7  }
8
```

関数 4

```
1  function uniq4(array) {
2    const knownElements = {};
3    const uniquedArray = [];
4    for (let i = 0, maxi = array.length; i < maxi; i++) {
5      if (array[i] in knownElements)
6        continue;
7      uniquedArray.push(array[i]);
8      knownElements[array[i]] = true;
9    }
10   return uniquedArray;
11 };
```

関数 5

```
1  function uniq5(array) {
2    const uniquedArray = [];
3    for (const elem of array) {
4      if (!uniquedArray.includes(elem))
5        uniquedArray.push(elem);
6    }
7    return uniquedArray;
8  }
```

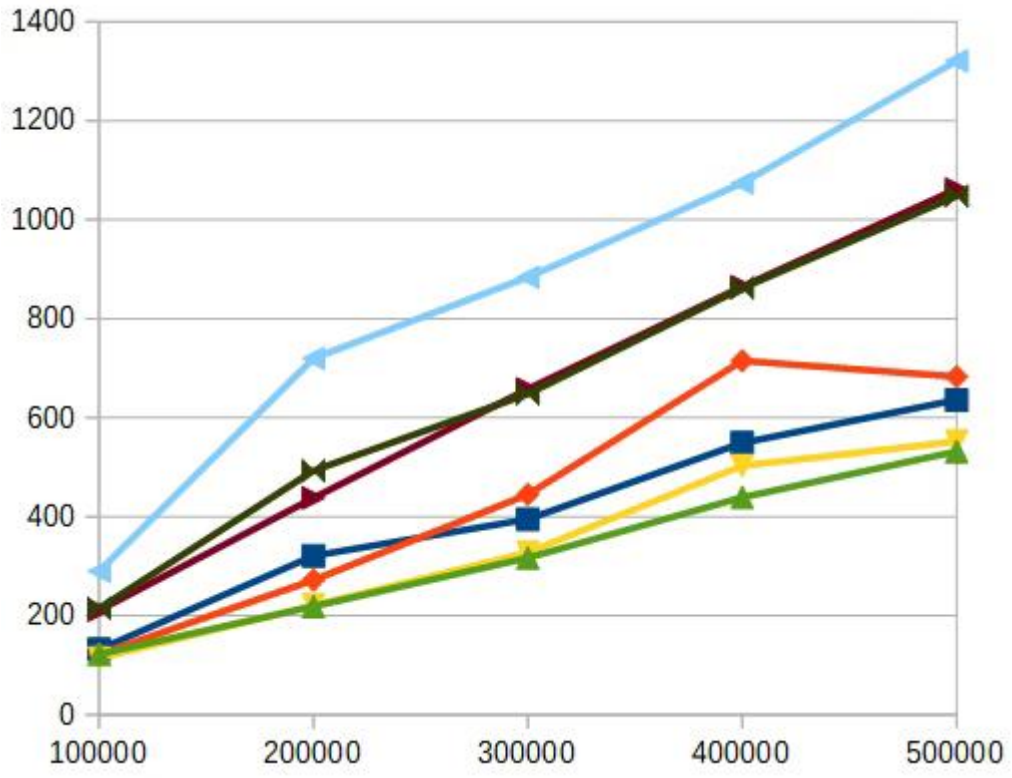
関数 6

```
1  function uniq6(array) {
2    const knownElements = new Map();
3    const uniquedArray = [];
4    for (const elem of array) {
5      if (knownElements.has(elem))
6        continue;
7      uniquedArray.push(elem);
8      knownElements.set(elem, true);
9    }
10   return uniquedArray;
11 }
```

練習課題 2

課題 1 の 6 つの関数はすべて、与えた配列から重複を取り除く JavaScript の関数です。

それぞれの関数について、性能を比較調査するために、要素数 300 の配列から重複を取り除く操作を 1 万回、2 万回、3 万回、4 万回、5 万回……と、1 万回ずつ増やしながら 10 万回まで実行して、以下のようなグラフで比較するのに必要な情報を得る方法を考えて、実装してみてください。



グラフを作るためには、以下のような表形式で表現できる情報が必要です。

実行回数	所要時間（ミリ秒）
10000	200
20000	400
30000	600
…	…

任意の処理の所要時間を計測する方法は、以下の要領です。

```
let startAt = Date.now();
（何らかの処理）
let delta = Date.now() – startAt;
// この「delta」が所要時間をミリ秒で表した数値となる
```

任意の個数の重複を含む配列を作成する処理は、以下の要領です。

```
function prepareArray(length) {
  const array = [];
  for (let i = 0; i < length; i++) {
    array.push(parseInt(Math.random() * (length / 10)));
  }
  return array;
}
```

JavaScript が不得手な方は、別の言語で実装してもよいものとします。
また、近くの他の受講者の方と協力して実装してもよいものとします。

練習課題 3

課題 1 の 6 つの関数について、要素数が増加したときの計算量の増大の仕方が、予想した O 記法の通りであるかどうかを実際に確認するために、必要な情報を得る方法を考えて、実装してみてください。

JavaScript が不得手な方は、別の言語で実装してもよいものとします。
また、近くの他の受講者の方と協力して実装してもよいものとします。

練習課題 4

課題 1 の 6 つの関数よりも計算量が少なくなるアルゴリズムがあるかどうかを考えて、実装してみてください。

JavaScript が不得手な方は、別の言語で実装してもよいものとします。
また、近くの他の受講者の方と協力して実装してもよいものとします。