2024年度　ソフトウェア工学概説

アプリケーション説明書

| 学籍番号 | 2220020 |
| --- | --- |
| 名前 | 伊藤豪 |
| アプリ名 | ソートアルゴリズムシミューレター |

**１．アプリの概要：**

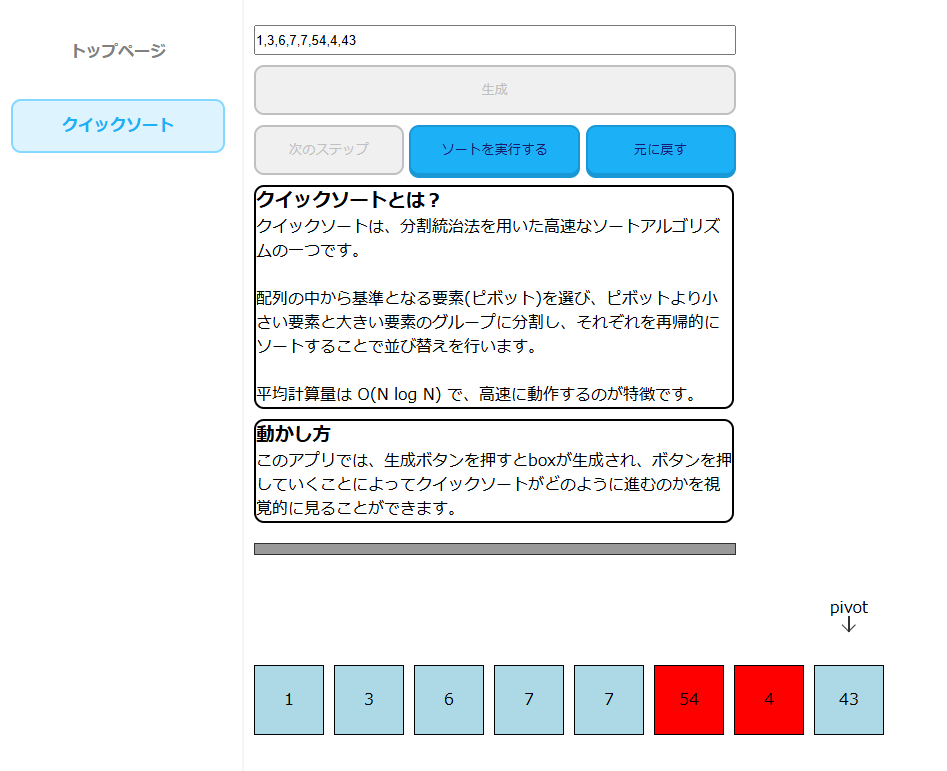
　このアプリは、ソートアルゴリズムの一つであるクイックソートをアニメーションによって可視化した状態で学習するツールである。

　ソートアルゴリズムについて学習したいユーザーがソートの過程を可視化した状態で見ることができる。

　各ステップでデータがどのように並べ替えられていくのかを視覚的に理解できるため、アルゴリズムの動作原理を深く学ぶことができる。

　ソートをする配列をユーザーによって自由に変えることができるため、いろんな場合にどのようにソートされていくのかを理解することができる。

**２．開発した機能：**



* **クイックソートの視覚化機能**

　ソートを実行するボタン、次のステップボタンを押すことによってクイックソートの過程を視覚的に理解できるようにした。

　数値の入ったボックスが並び、ソートの過程でボックスが動くことで、要素の比較や交換の流れを視覚的に理解できる。

　クイックソートにはピボットやボックスを交換するアニメーションが必要なためそれらについてわかりやすいようにピボットにはテキスト、交換するボックスは色を変更した。

* **入力値の変換**

　このアプリでは元々デフォルトの配列の値が入力されているがテキストエリアに新しく値をカンマ区切りで入力することによって新しい配列を画面上に表示させることができる。

　文字などを入れてしまった場合はうまく変換できないが、カンマを余分に入れてしまったりしたとしてもうまく配列に変換できるようになっている。

* **リセット機能**

「元に戻す」ボタンを押すことによって初期状態に戻すことができる。

何度でも異なるパターンでソートを試すことができるようになっている。

**３．アプリを作る上で工夫した点，特に評価して欲しい点：**

* **クイックソートの視覚化**

　クイックソートの処理過程をアニメーションでわかりやすく表現することに重点を置いた。

GSAP.jsと呼ばれるアニメーション制作を行う際に高機能なライブラリを活用し、スムーズな移動や交換アニメーションを実装した。

初めはCANVASの中をJavascriptによって動かす実装をしようと考えていたが、クイックソートのアニメーションでは何度も同じボックスを交換する必要があるため単純な交換の実装では難しいことからライブラリの採用をした。

要素の比較やスワップが視覚的に明確になるよう色の変化や動きにこだわった。

ステップごとに処理を確認できるようにすることによってアルゴリズムの流れを直感的に理解しやすくした。

* **ステップ実行機能による学習のしやすさ**

一度にソートを完了させるのではなく、1ステップずつ進められる機能を搭載した。

それによってクイックソートの「分割」、「交換」の流れを確認できる。

学習目的で利用する場合にどのように要素が動くのか理解できる。

手動で進めながら自分でアルゴリズムの動作を考えることができる。

* **大量データに対応することができる**

テキストエリアにどのような長さのデータを入力されたとしてもソートをすることができる。

少ないデータでも多いデータでも正しくクイックソートを動かすことができる。

* **UI・UXの工夫(操作のしやすさ)**

見た目のデザインや操作性にも配慮し、誰でも直感的に使えるように工夫した。

サイドバーを左側に配置することによって、メニューをわかりやすく配置した。

操作ボタンを中央に配置することによって視線の流れを考慮した。

動かせるボタンをホバー時の動きや影によって視認性を向上させ、動かないボタンについては視覚的にわかりやすくした。

押してほしいボタンのみを有効、押してほしくないボタンは無効にすることで説明書を読んでいないユーザーも操作できるようにした。

**４．アプリの信頼性：**

* どのようなテストを行ったか
  + ソートアルゴリズムの正確性

　クイックソートの実装が理論通りに動作するかを異なる入力データによっ　　　　　　て検証

　ソート後のデータが正しい順番になっているか確認

* + データ入力の処理

　カンマ区切りの入力を受け取ったとき、適切に数値配列に変換できるか

　　　　　　　空の入力、数値以外の入力が含まれたときにどのような処理をするか

* + UIボタンの動作

　アプリ内のボタンが期待通りの挙動をするか

* ソート処理開始から完了までの流れが問題なく進行するか
* アプリの信頼性について
  + モジュール化を意識することによって機能ごとにコードを分割した。
  + 後から増える可能性のある機能群に関してはファイルに分離することによって後から追加しやすくなるようにした
* 使用したコーディング規約
  + CSSのコードは<head>タグ内に書く
  + Javascriptのコードは<body>タグ内の最後に書く
  + Javascriptの変数の宣言にはlet,constを使用する
  + 変数、関数、id名等を命名する際は意味のわかりやすい英単語を利用する
  + 無名関数を宣言するときはインデントする
  + if,for文を使用するときはその中身をインデントする
  + 文末にはセミコロンを使用する
* 使用したコーディング規約(最終課題で指定された以外のコーディング規約)
  + 使用しない記述を残さない
    - 使用しない関数や変数を定義したまま使わない状態だと保守性を損なってしまうため使用しない記述は残さないようにした。
  + 宣言の書き方を統一する
    - 関数に関してはfunctionによって定義するようにし、関数の中で定義する変数や定数に関してはconstで定義している。
    - コーディングをする際にfunctionがついている部分は関数だとわかりやすくして可読性を向上させるため使用した。

**５．まとめと今後の課題・発展性：**

　今回のソフトウェア工学概説の最終課題では、ソートアルゴリズムの一つであるクイックソートの可視化を行うツールを開発した。

　ソートアルゴリズムの学習をする際には大抵は文章によって学習を行うため理解に苦しむ学生が多い。

　そのため、可視化することができれば直感的に理解できる可能性を向上することができるため開発することにした。

　開発した機能それぞれに対してテストを行うことによって、エラーが出てしまう箇所がないか検証し、修正した。

　コーディングをする際にコーディング規約を独自に設定して開発を行い、保守性や可読性を向上させる取り組みを行った。

　今後はクイックソート以外のソートアルゴリズムの実装をすることが求められると考えられる。また、既に実装しているクイックソートに関してもステップごとにソートを動かすことはできるが止めることはできないため改善の余地があると考えている。

**参考文献：**

（アプリの作成，説明書の執筆時に参考にした文献（Web含む）を列挙してください．）

* Javascript クイックソート

<https://qiita.com/may88seiji/items/a08504d472b5cec5f916>

* GSAP Homepage

<https://gsap.com/>

* IPA SEC BOOKS:ESCR Ver.3.0

<https://www.ipa.go.jp/archive/publish/secbooks20180629.html>