Dokumentation der Implementierung der Funktion MERGE

# Allgemeine Annahmen

Folgende Annahmen habe ich als Entwickler getroffen:

* Die übergebenen Intervalle sind nicht vorsortiert, sondern können in einer beliebigen Reihenfolge übergeben werden.
* Die Funktion MERGE erhält die Intervalle als Liste.
* Jedes Interval [x,y] ist ein Int32 Array mit zwei Elementen x und y, wobei das Programm auch prüfen muss, ob die einzelnen Intervalle valide sind und falls nicht, muss es entsprechend damit umgehen.
* x ist immer kleiner als y. Sollte das nicht der Fall sein, ist das Interval nicht valide und das Programm muss damit umgehen können.

# Entscheidungen über das Design der Anwendung

Folgende Entscheidungen habe ich als Entwickler getroffen:

* Um das Bauen und Ausführen der Anwendung so einfach wie möglich zu gestalten, habe ich mich dazu entschieden, die Anwendung in C# zu programmieren. Nach meiner Erfahrung ist das Aufsetzen einer Entwicklungsumgebung hier am einfachsten. Alles was man benötigt, um die Applikation auszuführen, ist eine Installation des .NetCore v3.1 Frameworks und Visual Studios. Näheres dazu in der README oder im nachfolgenden Kapitel.
* Auf ein Übergeben der Intervalle über ein grafische Benutzeroberfläche oder das Einlesen aus einer Datei oder über die Argumente in der Main-Methode des Programms, habe ich aus Zeitgründen verzichtet. Stattdessen werden die zu verarbeitenden Intervalle in der Main-Methode der Hauptanwendung oder in den Unit-Tests definiert.

# Bauen und Ausführen der Anwendung

Nachfolgend sind die Schritte aufgelistet, die durchgeführt werden müssen, um das Programm zu öffnen, zu bauen und auszuführen:

* Benötigt wird eine Installation des .NetCore3.1 Frameworks. Diese steht [hier](https://dotnet.microsoft.com/en-us/download/dotnet/3.1) zum Download bereit. Meistens ist diese Abhängigkeit bereits installiert. Sollte das Programm beim Ausführen aber eine entsprechende Fehlermeldung ausgeben, fehlt auf dem Zielsystem die .NetCore 3.1.
* Eine Installation der IDE Visual Studio. Diese ist [hier](https://visualstudio.microsoft.com/de/downloads/) zu finden.
* Das Versionsverwaltungswerkzeug Git muss auf dem Zielsystem installiert sein. Dieses kann [hier](https://git-scm.com/downloads) heruntergeladen werden, falls nicht vorhanden.

Sind alle Abhängigkeiten installiert, kann mithilfe von Visual Studio und Git das Repository geklont werden:

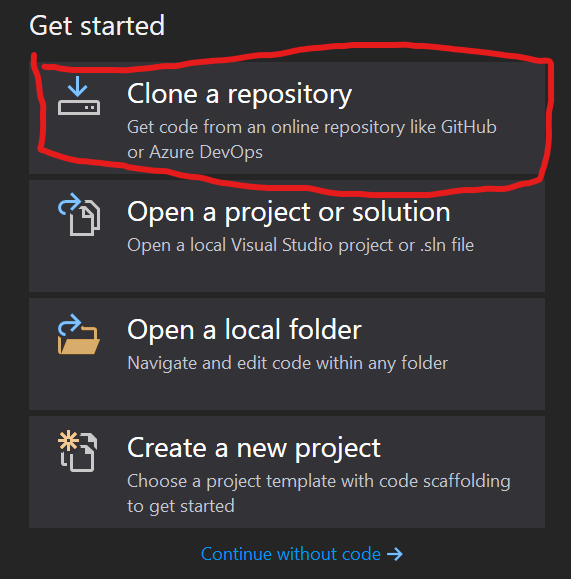


Abbildung 1: Klonen eines Repositories

Dabei öffnet sich sein neues Fenster, in welches die URL des Repositories sowie ein Zielordner eingefügt werden muss:

<https://github.com/chgsit00/interval-merge.git>

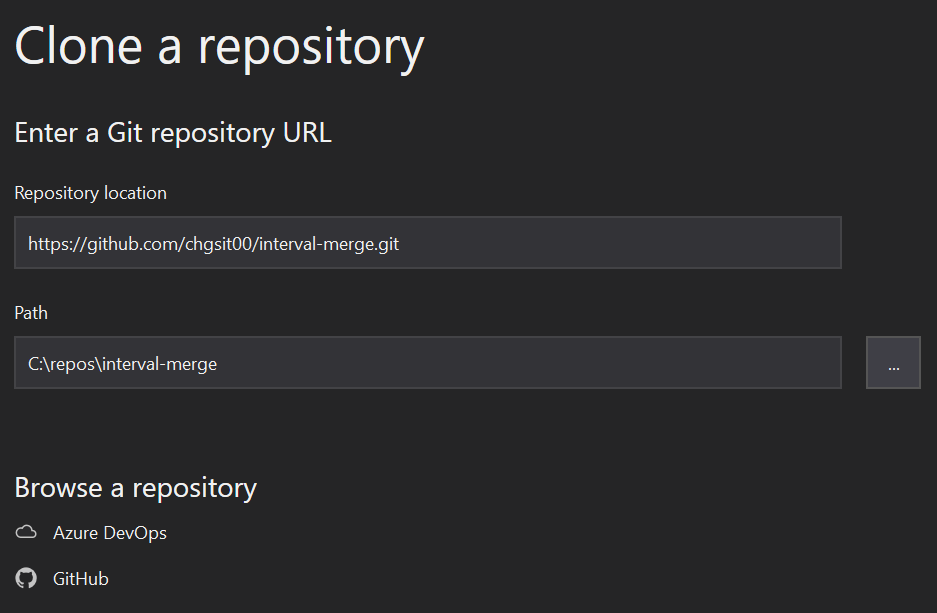


Abbildung 2: Einfügen des Links zum Repository

Nach Bestätigung der Eingaben, lädt Visual Studio die Solution, sowie die zugehörigen Projekte. Mithilfe des Run-Buttons in der oberen Leiste Visual Studio kann das Programm gebaut und ausgeführt werden:

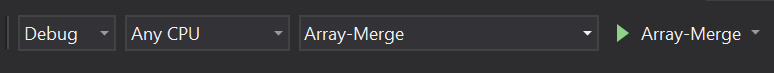


Abbildung 3: Run-Button

# Aufbau des Algorithmus

Bei der Implementierung des Algorithmus der Merge-Funktion waren mir folgende Punkte wichtig:

* Um die Komplexität des Codes niedrig zu halten und eine unnötige Beanspruchung des CPU zu vermeiden, sollten mehre verschachtelte for- oder while-Schleifen vermieden werden. Stattdessen sollte der Algorithmus nur ein einziges Mal über die Liste der übergeben Intervalle iterieren müssen.
* Um den Speicherbedarf niedrig zu halten, sollte der Algorithmus mit so wenig Kopien der Intervall-Liste wie möglich auskommen. Da diese beliebig lang sein kann, würden mehrere Kopien den RAM-Verbrauch in die Höhe treiben.
* Damit der Algorithmus nicht jedes Element der Liste mit jedem anderen Element der Liste vergleichen muss, sollte die Liste vorsortiert werden, sodass immer nur ein Listenelement mit dem vorherigen abgeglichen werden muss.

Daraus folgte folgender Algorithmus: