Minimumanforderungen für den Aufstieg in die 5. Klasse

Dieses Dokument fasst sehr grob jene Kenntnisse und Wissen zusammen, die du für den Aufstieg in die fünfte Klasse können solltest, um in SEW von Beginn an gut mitzukommen.

Ein ganz wichtiger Hinweis gleich zu Beginn: es hilft überhaupt nicht, die untenstehenden Themen theoretisch zu verstehen - bei SEW geht es immer um die Praxis. Du musst also wissen, wie du diese Themen in deinem Programm umsetzt und anwendest!

Basiswissen / Datenstrukturen

- Datentypen, Variablen setze ich voraus.
- Kontrollstrukturen wie Verzweigungen, Schleifen (for, while, foreach), Methoden (mit Parametern und Returnwerten) setze ich voraus.
- Strings und deren Funktionen (Teilstrings, Überprüfen ob ein String in einem anderen vorkommt, Zerteilen, Länge, Groß-/Kleinschreibung, leer/null Überprüfung, ...)
- Listen und Arrays (Einfügen, Löschen, Ersetzen, Durchsuchen)
- Dictionary (Einfügen, Überprüfen, Löschen von Einträgen); Dictionary von Listen (z.B. ein Dictionary, das zu einer ID eine Liste von Produkten/Noten/Personen/... speichert)
- Einlesen und Erstellen von CSV Dateien

Algorithmen / Business Logic

- Sehr oft kommt es vor, dass die Funktionalität (also die Business Logik) nicht implementiert werden kann
- Häufige Anforderungen: Suchen eines Elementes (größtes, kleinstes, mit einer bestimmten Id) in einer Liste, Filtern von Listen nach einem oder mehreren Kriterien, einfache Berechnungen (Summe der Preise einer Liste von Objekten, dann noch Rabatt, etc.)
- Siehe auch Arbeitsblatt *Algorithmische Übungen*

Objektorientierte Programmierung

- Klassen mit Feldern, Properties (full und auto), Konstruktoren (auch mehrere), Methoden, ToString
- Static vs. Non-Static, Klasse vs. Objekt
- Sichtbarkeiten (zumindest public, private, protected)
- Vererbung: abstrakte Klassen und Methoden, virtual/override, Basisklassenkonstruktor aufrufen, is-a Regel für die Vererbung
- Typtests mit 'is' bzw. 'as'
- Interfaces: Definition und Implementierung von Interfaces; Vererbung von Interfaces

WPF

- UI Controls: Label, TextBlock, TextBox, CheckBox, ComboBox, Button, RadioButton, ListBox, DataGrid, Image
    - Nur einige Spalten im DataGrid anzeigen, Anzeige in der Listbox anpassen
- UI Events: vor allem Click und SelectionChanged (auch: MouseDown, TextChanged, …)
- Data Binding mit ObservableCollection (event. auch INotifyPropertyChanged Interface)
- Richtige Verwendung von SelectedItem/SelectedIndex
- Mehrere Fenster verwenden und Daten zwischen ihnen austauschen
- Layout Container: StackPanel, DockePanel, Grid

LINQ

- Standardoperatoren: Where, Select, First/FirstOrDefault, Any, All, OrderBy/ThenBy, OrderByDescending/ThenByDescending
- Aggregate: Sum/Min/Max/Count/Avg
- Fortgeschrittene Operatoren: SelectMany, GroupBy, Join
- Extension Methods

Datenbankzugriff

- Ado.Net (connected mode)
    - Connection Strings, Verwendung von Connections, Commands
    - ExecuteReader/ExecuteNonQuery/ExecuteScalar,
    - Abfrage auf NULL
    - Parametrisierte Queries
- Entity Framework (EF) Core
    - Erstellen von Context und Entities (auch in unterschiedlichen Projekten) mit EF Core PowerTools
    - Einfügen/Ändern/Löschen von Einträgen, SaveChanges
    - Laden von abhängigen (related) Daten mit Include (auch mehrere Ebenen)
    - Zyklen im Ojektgraphen für die Serialisierung durchbrechen

Netzwerkprogrammierung

- Lokale und remote Endpunkte (endpoints) verstehen
- UDP Programmierung
    - Verwendung von UdpClient (am Client und am Server)
    - Umwandeln von Strings in Bytes (und umgekehrt)
- TCP Programmierung
    - Verwendung von TcpClient/TcpListener (am Client und am Server)

- Verwendung von NetworkStream, StreamReader/StreamWriter, BinaryReader/BinaryWriter zum Übertragen von Daten

Multi-Threading

- Starten von neuen Tasks (mit und ohne Parameter)
- Nachfolger (continuation) Tasks festlegen
- Fehlerbehandlung in Tasks
- Cancellation
- Multi-Threaded Programme: neue Tasks in WPF oder Konsolenanwendungen starten, jeden Request eines UDP/TCP Servers auf einem eigenen Thread verarbeiten
- Synchronisation (Locks, Monitor, Mutex, Semaphore)
- Asnyc/await: Regeln/Konventionen, async WPF Event-Handler, async Main

REST

- C# REST Service (Web Api) Projekt erstellen, Controller hinzufügen
- Routen und HTTP Verben richtig definieren
- Parameter aus der Route und aus dem Request Body
- ActionResult Ergebnisse mit Status Code und Payload
- Async Routen
- Statische Daten verwenden (da jeder Request einen neuen Controller erzeugt), Daten in Sessions speichern
- Routen vom Client richtig aufrufen (HttpClient Klasse) und Antworten deserialisieren
- Verwendung des JsonSerializers (auch für Lesen und Schreiben von JSON Dateien)

Git

- Erstellen von Repositories in GitHub, Klonen von Repositories
- Commit, Pull, Push im master/main Branch
- Umgang mit Git in Visual Studio/Visual Studio Code
- Umgang mit GitHub Classroom

Nochmals zur Wiederholung: alle diese Themen müssen in der praktischen Anwendung gekonnt werden! Also nicht nur Folien, Cheat Sheets, Tutorials lesen - ihr müsst Programme schreiben!