SML を使用した Todo リストアプリケーションの実装

学生番号: 09B23549
提出者: 中嶋 空偉
E-mail: pj9y6y4w@s.okayama-u.ac.jp

締切日: 2025 年 5 月 26 日 (月)

1　プログラムの概要

本レポートでは，Standard ML（SML）を使用して実装した Todo リストアプリケーションについて述べる．このアプリケーションは，タスクの管理，追加，更新，削除などの基本的な機能を提供し，授業で学んだ SML の機能を活用して実装されている．

2　作ろうと思った背景

新しいプログラミング言語を学ぶ際，私は常に HelloWorld プログラムを作成し，次に Todo リストプログラムを実装するという習慣がある．HelloWorld は基本的すぎるため，今回は Todo リストの実装に焦点を当てた．

SML は関数型プログラミング言語であり，パターンマッチングや再帰関数，高階関数などの特徴的な機能を持つ．これらの機能を活用して，実用的なアプリケーションを実装できるかという挑戦的な目標を設定した．

3　実装したい機能

本アプリケーションでは，以下の CRUD（Create, Read, Update, Delete）機能を実装する：

- Create: 新しい Todo アイテムの作成
- Read: Todo リストの表示，完了/未完了のフィルタリング
- Update: Todo アイテムの更新，状態の変更
- Delete: Todo アイテムの削除

4　SML プログラムへの実装方法

4.1　データ構造の設計

まず，Todo アイテムを表現するためのデータ型を定義した：

```
datatype status = Pending | Completed;
```

```
datatype todo = Todo of {
    title: string,
    description: string,
    dueDate: string,
    status: status
};

type todoList = todo list;
```

4.2 主要な関数の実装

4.2.1 作成機能

新しい Todo アイテムを作成する関数を実装した：

```
fun createTodo (title: string, description: string,
                dueDate: string) : todo =
    Todo {
        title = title,
        description = description,
        dueDate = dueDate,
        status = Pending
    };
```

4.2.2 表示機能

Todo リストを表示する関数を実装した：

```
fun displayList (list: todoList) : unit =
    let
        fun display (l: todoList, index: int) : unit =
            case l of
                [] => ()
              | x::xs => (
                    print ("[" ^ Int.toString index ^ "] ");
                    displayTodo x;
                    display (xs, index + 1)
                )
    in
        display (list, 0)
    end;
```

4.2.3 更新機能

Todo アイテムの状態を更新する関数を実装した：

```
fun updateStatus (list: todoList, index: int,
                  newStatus: status) : todoList =
    let
        fun update (l: todoList, i: int,
                    acc: todoList) : todoList =
            case (l, i) of
                ([], _) => rev acc
              | (Todo {title, description, dueDate, status}::xs, 0) =>
                    rev acc @ (Todo {
                        title=title,
                        description=description,
                        dueDate=dueDate,
                        status=newStatus
                    }::xs)
              | (x::xs, n) => update (xs, n-1, x::acc)
    in
        update (list, index, [])
    end;
```

4.2.4 削除機能

Todo アイテムを削除する関数を実装した：

```
fun removeTodo (list: todoList, index: int) : todoList =
    let
        fun remove (l: todoList, i: int,
                    acc: todoList) : todoList =
            case (l, i) of
                ([], _) => rev acc
              | (x::xs, 0) => rev acc @ xs
              | (x::xs, n) => remove (xs, n-1, x::acc)
    in
        remove (list, index, [])
    end;
```

5 テスト結果

プログラムの動作確認を行った結果，以下の機能が正常に動作することを確認した：

- Todo アイテムの作成と表示
- 状態の更新（完了/未完了の切り替え）
- アイテムの削除
- 完了済み/未完了のフィルタリング

テストの実行結果は以下の通りである：

```
=== すべての Todo ===
[0] タイトル: テスト
説明: テスト用の Todo
期限: 2024-03-21
状態: 完了
------------------
[1] タイトル: テスト 2
説明: 2 つ目のテスト
期限: 2024-03-22
状態: 未完了
------------------

=== 完了済みの Todo ===
[0] タイトル: テスト
説明: テスト用の Todo
期限: 2024-03-21
状態: 完了
------------------

=== 未完了の Todo ===
[1] タイトル: テスト 2
説明: 2 つ目のテスト
期限: 2024-03-22
状態: 未完了
------------------
```

6 まとめ

本レポートでは，SML を使用した Todo リストアプリケーションの実装について述べた．授業で学んだ SML の機能を活用し，実用的なアプリケーションを実装することができた．特に，パターンマッチングや再帰関数，高階関数などの機能を効果的に使用することで，簡潔で保守性の高いコードを実現することができた．授業で習った時は，シンプルなものしか作れないと感じていたが，実用的なアプリケーションを実装することができたことに満足している．

今後の課題として，以下の機能の追加が考えられる：

- 期限によるソート機能
- ファイルへの保存/読み込み機能
- より詳細な検索機能
- エラー処理の強化