

Използвайте операциите `map`, `foldr`, `foldl`, `filter` и реализация на списък по ваше желание, за да решите следните задачи:

1. Намерете максималния елемент на списък от цели числа
2. Намерете минималния четен елемент на списък от цели числа
3. Намерете най-големия елемент, по-малък от определено число в сортиран списък
4. Намерете най-малкия елемент, по-голям от определено число в сортиран списък
5. Намерете сумата на най-големите делители на списък от цели числа
6. По предварително зададен списък от обекти от тип `FoodItem` и зададен списък от цели числа, които представляват `id`-та на обекти от тип `FoodItem`, намерете сумата на цените на реферираните чрез `id` обекти
7. По зададен списък от обекти от тип `FoodItem`, намерете колко пъти се среща всеки от обектите (ако обектите имат едно и също `id`, приемаме че са един и същи обект)
8. По зададен списък от обекти от тип `FootballGame` създайте обект `Leaderboard`, който да отразява коректно резултата от изиграните мачове
9. Реализирайте класа `UndoRedoStringBuilder` чрез `DLL`
10. Реализирайте програма, която да дава възможност на 4-ма човека да играят на играта "Монополи" (за простота, нека полетата са само обекти за закупуване, без карти шанс)

Класът `FoodItem` има следните данни като минимум:

- `Id` - цяло число, уникален идентификатор на обекта
- `Price` - число с плаваща запятая, цена на обекта
- `Name` - стринг, име на обекта
- `Weight` - число с плаваща запетая, тегло на обекта
- `barCode` - стринг, баркод на обекта

Класът `FootballGame` има следните данни като минимум:

- `homeTeam` - обект от тип `FootballClub`, описващ домакина на срещата
- `awayTeam` - обект от тип `FootballClub`, описващ госта в срещата
- `result` - `char` { '1' - домакинска победа(3 точки за `homeTeam`), 'X' - равен(по една точка за двата отбора), '2' - победа на гостите(3 точки за `awayTeam`) }

Класът `FootballClub` има следните данни като минимум:

- `Id` - цяло число, уникален идентификатор на футболния клуб
- `Name` - стринг, име на футболния клуб

Класът `Leaderboard` има следните данни като минимум:

- `competitionName` - стринг, име на надпреварата
- `teamResults[]` - списък от тип `TeamResult`

Класът `TeamResult` има следните данни като минимум:

- `Team` - обект от тип `FootballClub`

- Points - цяло число, точки, събрани от отбора
- gamesPlayed - цяло число, брой изиграни срещи
- Wins - цяло число, брой победи
- Losses - цяло число, брой загуби
- draws - цяло число, брой равни мачове

Sample foldl function:

...

```
T foldl(std::function<T(T, T)> func, T initialValue) {
    return foldlHelper(head, func, initialValue);
}
```

private:

```
// Helper function for foldr
T foldlHelper(Node<T>* node, std::function<T(T, T)> func, T initialValue) {
    if (!node) {
        return initialValue; // Base case: return the accumulated value
    }
    // Recursive case: fold the rest of the list and apply the function
    return func(node->data, foldlHelper(node->next, func, initialValue));
}
```

...

```
LinkedList<int> list;
```

```
// Adding elements to the linked list
```

```
list.append(1);
```

```
list.append(2);
```

```
list.append(3);
```

```
// Using foldr to sum all elements in the linked list
```

```
int sum = list.foldl([](int x, int y) { return x + y; }, 0);
```