Rückblick auf Ihre Lösungen

Was ist an dieser Schleife denn „unschön“?

```
for (int i = 0; i < seq.size() - 2; i++)
{
    if (seq[i] == "thousand" && seq[i + 2] == "hundred")
            {
            seq[i] = "";
            }
}
```

Was ist an dieser Schleife denn „unschön“?

```
for (int i = 0; i < seq.size() - 2; i++)
{
    if (seq[i] == "thousand" && seq[i + 2] == "hundred")
            {
            seq[i] = "";
            }
}
```

Wie oft läuft die Schleife, wenn seq 0 oder 1 Element enthält?

Was ist an dieser Schleife denn „unschön“?

```
for (int i = 0; i < static_cast<int>(seq.size()) - 2; i++)
{
    if ("thousand"  == seq[i]  &&   "hundred“ == seq[i + 2])
                {
                seq[i] = "";
                }
}
```

Teil 2

Was ist an dieser Schleife denn „unschön“?

```
auto iter = ..... /* irgendwas */
for (int i = aktW;    seq[i] != " ";    i++)
 {
      if (seq[i] == iter) { …
```

Was ist an dieser Schleife denn „unschön“?

```
auto iter = ..... /* irgendwas */
for (int i = aktW;    seq[i] != " ";    i++)
 {
      if (seq[i] == iter) { …
```

Hier liegt keine for-Schleife vor!!!

```
int i=aktW;
while (i < seq.size() &&
          ! seq[i].empty())
{
    if (seq[i] == iter) { …
```

Teil 3

Was ist „unschön“?

```
#include <string>
#include <ctime>
#include <vector>
class AtcoCommand {
private:
    std::string* p_fileName = nullptr;
    tm* p_dateTime = nullptr;
    std::string* p_wordSequence = nullptr;
    vector<std::string>* p_commands = nullptr;
public:
   AtcoCommand();
   AtcoCommand(const AtcoCommand &obj);
   ~AtcoCommand();
/* …   */
```

Was finden Sie gut, was kann man verbessern

Was ist „unschön“?

```
#include <string>
#include <ctime>
#include <vector>
class AtcoCommand {
private:
    std::string* p_fileName = nullptr;
    tm* p_dateTime = nullptr;
    std::string* p_wordSequence = nullptr;
    vector<std::string>* p_commands = nullptr;
public:
   AtcoCommand();
   AtcoCommand(const AtcoCommand &obj);
   ~AtcoCommand();
```

+ Alles Nötige eingebunden
+ kein using namespace std
+ Zeiger beginnen mit „p_“
+ Initialisierung

- Vor vector gehört auch std::vector
- Wertesemantik verwenden, keine Notwendigkeit für Zeiger

Verbesserung

```
#include <string>
#include <ctime>
#include <vector>
class AtcoCommand {
private:
    std::string m_fileName;
    tm m_dateTime;
    std::string m_wordSequence;
    std::vector<std::string> m_commands;
public:
   AtcoCommand();
   AtcoCommand(const AtcoCommand &obj);
   ~AtcoCommand();
```

- Vor vector gehört auch std::vector
- Wertesemantik verwenden, keine Notwendigkeit für Zeiger

Teil 4

Was ist „unschön“?

```
class AtcoCommandEvaluator {
private:
std::string _expectedTypesFileName;
/* ..*/
public:
    AtcoCommandEvaluator(const std::string&
      expectedTypesFile, std::unordered_set<std::string>
      fileNames);
    std::string getExpectedTypesFileName();
    std::string getTopTenWords();
    std::string getTopCommand();
    std::string getReadFiles();
    std::string getExpectedCommands();
```

Was finden Sie gut, was kann man verbessern

Was ist „unschön“?

```
class AtcoCommandEvaluator {
private:
std::string _expectedTypesFileName;
/* ..*/
public:
    AtcoCommandEvaluator(const std::string&
      expectedTypesFile, std::unordered_set<std::string>
      fileNames);
    std::string getExpectedTypesFileName() const;
    std::string getTopTenWords() const;
    std::string getTopCommand() const;
    std::string getReadFiles() const;
    std::string getExpectedCommands() const;
```

Die Getter sollten const sein

Teil 5

Was ist „unschön“?

```
class AtcoCommand
{
    /*  … */
public:
    AtcoCommand();
    std::string &   getFileName();
    void setFileName(const std::string& fileName);
    tm& getDateTime();
    std::string& getWordSequence();
```

Was finden Sie gut, was kann man verbessern

Was ist „unschön“?

```
class AtcoCommand
{
    /*  … */
public:
    AtcoCommand();
    const std::string &   getFileName()  const;
    void setFileName(const std::string& fileName);
    const tm& getDateTime() const;
    tm& getDateTimeForMod();
    const std::string& getWordSequence() const;
```

const verwenden