Swinburne University of Technology

School of Science, Computing and Emerging Technologies

MIDTERM COVER SHEET

Subject Code:	COS30008
Subject Title:	Data Structures and Patterns
Assignment number and title:	Midterm Project: Solution Design & Iterators
Due date:	April 16, 2025, 18:59
Lecturer:	Dr. Markus Lumpe

Your name: Dhanveer Ramnauth **Your student ID:** 103866373

Marker's comments:

Problem	Marks	Obtained
1	64	
3	196	
Total	260	

Figure 1: AutoKey.cpp

```cpp
#include "AutoKey.h"

AutoKey::AutoKey(const std::string& aKeyword) noexcept
    : fValue(), fKeyLength(0), fIndex(0)
{
    for (char c : aKeyword)
    {
        if (std::isalpha(c))
            fValue += std::toupper(c);
    }
    fKeyLength = fValue.size();
}

size_t AutoKey::size() const noexcept
{
    return fValue.size();
}

char AutoKey::operator*() const noexcept
{
    return fValue[fIndex];
}

AutoKey& AutoKey::operator++() noexcept
{
    if (fIndex < fValue.size()) {
        fIndex++;
    }
    return *this;
}

AutoKey AutoKey::operator++(int) noexcept
{
    AutoKey temp = *this;
    ++(*this);
    return temp;
}

AutoKey& AutoKey::operator+=(char aChar) noexcept
{
```

```
41      if (std::isalpha(aChar)) {
42          fValue += std::toupper(aChar);
43      }
44      return *this;
45  }
46
47  void AutoKey::reset() noexcept
48  {
49      fValue = fValue.substr(0, fKeyLength);
50      fIndex = 0;
51  }
```

Figure 2: VigenereIterator.cpp

```cpp
#include "VigenereIterator.h"

VigenereIterator::VigenereIterator(const std::string& aKeyword,
                                   const std::string& aSource,
                                   EVigenereMode aMode) noexcept
    : fMode(aMode),
      fKeys(aKeyword),
      fSource(aSource),
      fIndex(0),
      fCurrentChar(0)
{
    initializeTable();
    if (fMode == EVigenereMode::Encode)
        encodeCurrentChar();
    else
        decodeCurrentChar();
}

char VigenereIterator::operator*() const noexcept
{
    return fCurrentChar;
}

VigenereIterator& VigenereIterator::operator++() noexcept
{
    fIndex++;
    if (fMode == EVigenereMode::Encode)
        encodeCurrentChar();
    else
        decodeCurrentChar();
    return *this;
}

VigenereIterator VigenereIterator::operator++(int) noexcept
{
    VigenereIterator old = *this;
    ++(*this);
    return old;
}

```

```cpp
bool VigenereIterator::operator==(const VigenereIterator& aOther)
    const noexcept
{
    return fIndex == aOther.fIndex
       && fSource == aOther.fSource;
}

VigenereIterator VigenereIterator::begin() const noexcept
{
    VigenereIterator temp = *this;
    temp.fIndex = 0;
    temp.fKeys.reset();

    if (temp.fMode == EVigenereMode::Encode)
        temp.encodeCurrentChar();
    else
        temp.decodeCurrentChar();

    return temp;
}

VigenereIterator VigenereIterator::end() const noexcept
{
    VigenereIterator temp = *this;
    temp.fIndex = fSource.size();
    return temp;
}

void VigenereIterator::encodeCurrentChar() noexcept
{
    char c = fSource[fIndex];
    fCurrentChar = c;
    if (!std::isalpha(c)) return;

    bool isLower = std::islower(c);
    char upperC = std::toupper(c);
    char keyChar = *fKeys;

    char encoded = fMappingTable[keyChar - 'A'][upperC - 'A'];
    fCurrentChar = isLower ? std::tolower(encoded) : encoded;
```

```
80
81      fKeys++;
82      fKeys += upperC;
83  }
84
85  void VigenereIterator::decodeCurrentChar() noexcept
86  {
87      char c = fSource[fIndex];
88      fCurrentChar = c;
89      if (!std::isalpha(c)) return;
90
91      bool isLower = std::islower(c);
92      char upperC = std::toupper(c);
93      char keyChar = *fKeys;
94
95      char decoded = 'A';
96      for (size_t i = 0; i < CHARACTERS; i++)
97      {
98          if (fMappingTable[keyChar - 'A'][i] == upperC)
99          {
100             decoded = 'A' + i;
101             break;
102         }
103     }
104
105     fCurrentChar = isLower ? std::tolower(decoded) : decoded;
106
107     fKeys++;
108     fKeys += decoded;
109 }
```