Exercise 1:
The Commandments of Software Engineering

Hand-out: 2 April 2004
Due: 16 April 2004

Please solve this exercise alone.

1. Summary: Modularity principles

Uniform Access principle
Facilities managed by a module are accessible to its clients in the same way whether implemented by computation or by storage.

Information Hiding principle
The designer of every module must select a subset of the module’s properties as the official information about the module, to be made available to authors of client modules.

Open-Closed principle
Modules should be open (may be extended) and closed (usable by clients).

2. Uniform Access principle
Here is a possible implementation of a class BankAccount in Java:

```java
import java.util.*;

class BankAccount {
    ...
    public int balance() {
        int depositSum = 0;
        int withdrawalSum = 0;

        for (int i = 0; i < deposits.size(); i++) {
            depositSum = depositSum + ((Integer)deposits.get(i)).intValue();
        }
    }
}```
```java
for (int i = 0; i < withdrawals.size(); i++) {
    withdrawalSum = withdrawalSum +
                   ((Integer) withdrawals.get(i)).intValue();
}
return depositSum - withdrawalSum;
```

To do

- Does this style observe the principle of *Uniform Access*?
- What are the benefits of applying the *Uniform Access* principle?

Hint

Examine where you need parentheses in Java.

To hand in

Hand in your answers to the two questions above (in the “To do” section).

### 3. Information Hiding principle

Here is a bad application written in Java:

```java
class BadApplication {
    ....
    public void changeBalance() {
        account.balance = 1000;
    }
    ....
    protected BankAccount account;
}
```

using the following class `BankAccount`:

```java
class BankAccount {
    ....
    public int balance;
}
```

To do

- Why does this example violate the principle of *Information Hiding*?
- How would you solve the problem?
- Write the equivalent example in Eiffel, applying standard style rules.
Hint
The Eiffel code corresponding to the above Java classes would not compile.

To hand in
Hand in your answers to the two questions above (in the “To do” section) and the text of your Eiffel classes APPLICATION and BANK_ACCOUNT.

4. Open-Closed principle

To do
Give an example showing the importance of the Open-Closed principle. (Explain the benefits of this principle.)

Hints
• Think about inheritance and client relationships.
• You may view the question as: what would not be possible without this principle?

To hand in
Hand in the example and accompanying explanations.