ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Programming Techniques for Scientific Simulations Exercise 4

HS 12
Prof. M. Troyer

Problem 4.1 Operator overloading and template functions
(no block assignment)

This exercise focuses on the implementation of a new type describing the $\mathbb{Z}_2$ group, and on the implementation of a generic power function working with all standard numeric types, as well as our new $\mathbb{Z}_2$.

Definition The finite group $\mathbb{Z}_2$ has the following properties:

- Group's element are: $\mathbb{Z}_2 = \{+, -\}$, where $+$ is the identity element.
- The group operation $\cdot$ is defined through:

$$
\begin{aligned}
+ \cdot + = - \cdot - = + \\
- \cdot + = + \cdot - = -
\end{aligned}
$$

- The representation of a group element $g(\mu)$, $\mu \in \mathbb{Z}_2$ on integer, real or complex numbers is given by

$$
g(\mu) = \begin{cases} +1, & \text{for } \mu = + \\ -1, & \text{for } \mu = - \end{cases}
$$

a) Implementation of $\mathbb{Z}_2$ To represent the $\mathbb{Z}_2$ group in C++, we will use the enumeration type.

```
enum Z2 { Plus, Minus };
```

The group operation will be implemented by overloading the * operator, i.e. we assign the correct meaning to the expression

```
Z2 p = Plus, m = Minus;
Z2 r = p*m;
```

Every time we make use of the operator *, the C++ compiler is looking for the function `operator*` with the correct types to be invoked[1]. In our case, we need to define:

```
Z2 operator*(Z2 a, Z2 b);
```

Furthermore, we want to be able to print our result in a nice form, i.e. using an expression such as `std::cout << r << std::endl;`. We will therefore overload

```
ostream& operator<<(ostream& os, Z2 a);
```

in such a way that *Plus* is printed for + and *Minus* for −.
To implement the action of a group element on a number, we will implement the following template function (note that from the point of view of C++, `a*b` is not necessarily the same as `b*a`):

```
template<class T> T operator*(T a, Z2 b);
template<class T> T operator*(Z2 a, T b);
```

[1] The same is also valid for all operators: $+, -, (), [], <<$, etc.

b) Implementation of generic power function We also want to implement a templated power function which only relies on the multiplication, so that it can also be used on our $\mathbb{Z}_2$ group:

```
template<class T> T mypow(T a, unsigned int n);
```

Hint: in order to be generic we provide a templated function `identity_element`, which returns one for all numeric types. How can you overload such a function, in order provide the identity element of the $\mathbb{Z}_2$ group?

Exercise
For this problem your tasks are:

1. Complete the skeleton code `z2_group_skeleton.cpp` we provide on the lecture homepage.
2. For each of the templated functions, think about the concepts required and document these.
3. Complete the skeleton code `z2_group_class_skeleton.cpp` with implementation of the group $\mathbb{Z}_2$ within a class.

Problem 4.2 Class Header for Animal and Genome (Block C)

Note: This part will be discussed in detail during the exercise class.
If you have not understood the Penna Model yet, please read the paper as suggested in Exercise 3.2.
Think of an animal and a genome as objects and write the class headers[2] for them.
At the point, **DO NOT IMPLEMENT ANY FUNCTION!** During next week exercise class we will organize personal discussions with the teaching assistants to comment your code *(this time you're not going to receive any comments by email).*

[2] The class header usually contains only the definition of the class (contains declaration of members).