Tutorial: Polymorphism

Song Liu (song.liu@bristol.ac.uk)
GA 18, Fry Building,
Microsoft Teams (search "song liu").

Polymorphism

1. In this tutorial, we will have a quick look at another important OOP feature: Polymorphism.
2. Polymorphism means functions that perform differently but share the same interface.

Motivation

1. Create a new class, called `symmmatrix` storing a symmetric matrix.

```
class symmmatrix: public matrix{

};
```

2. Symmetric matrix "is a" matrix. Thus the inheritance makes logical sense.
3. It inherits all the code I have written for the `matrix` class.

Motivation

1. `symmmatrix` also inherits the `set_elem` function.
2. We know that for a symmetric matrix, $A_{i,j} = A_{j,i}$.
3. However, nothing stops a irresponsible programmer from doing the following

```
A.set_elem(1,0, 1);
A.set_elem(0,1, -1); // A is not symmetric anymore!
```

4. `set_elem` should maintain the symmetry of our matrix!
5. We are in a situation that we want the child class to **share the same interface with its parent class** (`set_elem`), but **perform the task in a different way**.

Polymorphism

1. In fact, we can consider a symmetric version of `set_elem`.
2. When it is called, it sets both `i,j` and `j,i`-th element to `val`.
    - Hence, the symmetry is preserved.
3. Write a new `set_elem` for `symmmatrix` class.

```
class symmmatrix: public matrix{
    void set_elem(int i, int j, int val){
        // TODO: write your new set_elem function.
    }
};
```

Polymorphism

1. One more thing, you need to write a new constructor function for the `symmmatrix` class.
    - Recall: Constructors are not inherited.
2. Note that a symmetric matrix can only be a square matrix.
    - `num_cols` must be the same as `num_rows`.
    - Thus, your constructor should take only one input, indicating the number of rows **and** the number of columns.

Test

Now, let us test your `symmmatrix` class.

```
int main()
{
    matrix A(2, 3);
    symmmatrix B(3);

    for (int i = 0; i < 2; i++){
        for (int j = 0; j < 3; j++){
            // set i,j-th element to some random value.
            A.set_elem(i, j, rand() % 10);
        }
    }

    for (int i = 0; i < 3; i++){
        for (int j = 0; j < 3; j++){
            B.set_elem(i, j, rand() % 10);
        }
    }

    A.print();

    printf("------------------\n");

    B.print();

}
```

Test

1. What is the output of this program?
2. Does that match what your expectation?