Task for OOP

deadline: Jan 9, 2012, 11:00 Uhr

---

11. Design and implement your own class `MyVector` for a dynamic vector from the scratch, i.e., without using available STL-containers or libraries, such that the following code can be executed:

```
1   int main()
2   {
3    MyVector aa(5);
4    for (int i=0; i<aa.size(); i++)   aa.Put(i,(i+1)*3.14);
5
6    aa[3] = 4.5;
7    cout << "aa: " << aa << endl;
8
9    const MyVector bb(aa);
10         MyVector cc, dd;
11
12   cc = bb + aa;
13   dd = bb + 2*aa;
14   cout << "dd: " << dd << endl;
15
16   cout << dd.Get(5);
17
18   return 0;
19  }
```

    - You have to store the vector values in a dynamic array (member of class).
    - It is necessary to implement all needed constructors, the destructor (free the memory!) and the assignment operator.
    - Catch the bad_alloc exception in the main function if the memory allocation fails, see the tutorial.
      Increase the vector size such that you experience that error in line 13, 12, 9 and 3 (in that order).
    - Throw a range_error in the `Put-` and `Get-` methods whenever an index is out of range.
    - Implement the arithmetic operators.
    - Try also the new features of C++11 with `&&` (rvalue reference) in combination with move-constructor.

12. Implement the cg-method without preconditioning for solving the system of equations

$$Ax = b$$

    with a symmetric, positive definite square matrix $A_{n\times n}$, right hand side b and solution vector x. For our tests we use the Hilbert matrix which is a dense matrix.

    The algorithm has to be implemented using the boost-library (available in standard C++), especially by means of the included Basic Linear Algebra package uBLAS. Use the operators and function provided therein!!

    Hint: Have also a look at the vast number of available libraries in boost.

    ($*$) Funktioniert dies auch mit `compressed_matrix` oder `sparse_matrix`, siehe web?

13. Extend your vector class from exercise 11 by using expression templates such that the following operations are transformed at compile time (deadline: Jan 23, 2012).

```
1   int main()
2   {
3    MyVector aa(5),bb(5), cc(5), dd(5), ee(5);
4    double alfa(-1.5);
5    // initialize vectors;
6    ...
7
8    cc = aa + alfa*bb;
9    dd = alfa*aa + bb;
10   ee = aa/bb;                  // elementwise
11
12   dd += cc - aa/alfa;
13
14   return 0;
15  }
```

Have a look at the wiki entry for expression templates. Try the appropriate code (colored) and apply the framework to your own vector class.