Examination, Programming, bridging course, 2011-12-19  
Time, 14.00-19.00

Materials allowed: None

Each problem is worth 5 points.

For grade 4:

C++ students: Do problem 1, 3 and 5  
Java students: Do problem 1, 3 and 7  
Fortran students: Do problem 1, 3 and the grade 4 version of 9 and 10

For grade 5:

C++ students: Do problem 1, 2, 3, 4, 5 and 6  
Java students: Do problem 1, 2, 3, 4, 7 and 8  
Fortran students: Do problem 1, 2, 3, 4 and the grade 5 version of 9 and 10

To get the grade 4 you should complete 2/3 of the problems marked for grade four. To get the grade 5 you must complete 2/3 of all the problems.

If you achieve neither, you will get grade 3.
1. **Python**

   (a) What is the yield statement doing? Create a code example where it is used correctly.

   (b) Give two key ideas behind the design philosophy of the Python language.

   (c) Concerning Python objects, explain identity, type and value.

2. **Python**

   Write Python code to print the numbers 1 to 100, but every third number should be replaced with 'Fizz' and every fifth with 'Buzz'. A number which fulfills both conditions should be replaced with "Fizz Buzz". The start of the output should be as follows:

   1, 2, Fizz, 4, Buzz, Fizz, 7, 8, Fizz, Buzz, 11, Fizz, 13, 14, Fizz Buzz, 16, 17, Fizz, 19, Buzz, ...

   Exact formatting does not matter, but the result should be clearly readable.
3. C

The **Golden section** is a classical way to divide a line segment into two pieces to achieve the **Golden ratio**.

If we divide the line segment into two pieces of the length \( a \) and \( b \), where \( a \) is greater than \( b \) then the following should be true to have a **Golden ratio**.

\[
\frac{a + b}{a} = \frac{a}{b}
\]  \hspace{1cm} (1)

Example:
If the length of the line segment is 1.2 meters and we divide it into two pieces, one of the approximate length 0.742 and the other of length 0.458 the equation above is satisfied, thus we have a **Golden ratio**.

Task:
Write a C-function with two parameters, the total length of the line segment and the length of the longest sub segment (called \( a \) above), that tests whether we have a **Golden ratio** or not. The function should return the outcome of the test.

Also write a main function that tests the function with some values.

4. C

An accurate approximation of the \( \sin(x) \) function around \( x=0 \) is the following

\[
\sin(x) = \sum_{n=1}^{\infty} \frac{(-1)^{n-1}x^{2n-1}}{(2n-1)!} = x - \frac{x^3}{3!} + \frac{x^5}{5!} \cdots 
\]  \hspace{1cm} (2)

Task:
Write a function `double xsin(double x)` that calculates and returns the approximate value of \( \sin(x) \) using the formula above. Stop the summation when the term to be added is absolutely less that a desired epsilon (small number).

Also write a main function that asks for a value of \( x \) and uses the function above to calculate and print the approximate value of \( \sin(x) \). Also compute and print the value of \( \sin(x) \) using the C built in function.
5. C++

Assume that you have the following class definition in C++

class Container {
private:
    int *buffer; // data storage
    int size; // size of the buffer
    int number; // actual use, always <= size
public:
    Container(); // constructor
    Container(int size); // constructor
    ~Container(); // destructor
    void store(int data); // store data in the first free // place
    int find(int index); // return the data in position 'index'
};

Thus we have a class that can store a number of integer values in the array buffer. It has a predefined size that we set in the constructor. The attribute size shows this size. When storing numbers we start at the first position, then the second position and so on, There is no way to remove data. The attribute number shows the first free position.

The method store should store the integer data, (its parameter) in the container at the first free position. If the container is full, it should be expanded to twice the size.

The method find returns the value stored in the position index (its parameter). It must check that we give a valid index and handle invalid requests somehow.

Task:
Implement all of this class in C++ as described above.

6. C++

Below is a declaration of a template class, VectorField, and two other classes, Matrix and Vec, that it depends on. A vector field is similar to a matrix, but instead of each element having one value, it is a vector, a number of values.

template <class T, int N>
class Vec
{
public:
inline T& operator[](int i);
T p_norm(int p);

protected:
  T data[N];
};

template <class T>
class Matrix
{
public:
  Matrix();
  Matrix(int nWidth, int nHeight);

  // Operator to index one element, example: mat(x,y) = 1;
  T& operator ()(int i, int j);

protected:
  int mnWidth;
  int mnHeight;
  vector<vector<T>> mData;
};

template<class T, int N>
class VectorField : public Matrix<Vec<T,N>>
{
public:
  VectorField();
  VectorField(int nWidth, int nHeight);

  void randomize();

  Matrix<T> p_norm(int p);
};

Task: Implement a function to calculate the p norm of each vector in the vector field. The result of this operation will be a standard matrix with the same dimensions as the vector field. Implement this in the p norm-functions in Vec and VectorField. It is suitable to call the p norm function in Vec from the one in VectorField. The p-norm for a vector is given by the formula

$$\|x\|_p := \left( \sum_{i=1}^{n} |x_i|^p \right)^{1/p}.$$  

You can assume the other functions in the classes are already implemented. Note:
'vector' above refers to a mathematical vector, which in this case is implemented using the Vec class instead of vector from STL.
7. Java

Assume that you have the following class definition in Java:

```java
public class Container {
    private int[] buffer; // data storage
    private int size; // size of the buffer
    private int number; // actual use, always <= size

    public Container(){} // constructor
    public Container(int size){} // constructor
    public void store(int data){} // store data in the first free place
    public int find(int index){} // return the data in position 'index'
}
```

All methods and constructors are left empty here, they should of course be filled with some code.

Thus we have a class that can store a number of integer values in the array `buffer`. It has a predefined size that we set in the constructor. The attribute `size` shows this size. When storing numbers we start at the first position, then the second position and so on. There is no way to remove data. The attribute `number` shows the first free position.

The method `store` should store the integer `data`, (its parameter) in the container at the first free position. If the container is full, it should be expanded to twice the size.

The method `find` returns the value stored in the position `index` (its parameter). It must check that we give a valid index and handle invalid requests somehow.

Task:
Implement all of this class in Java as described above, ie implement all constructors and methods.

8. Java

Expand the problem in problem 7 so that you also have two methods:

```java
    public int remove(int index){}
    public void sort() {}
```
The first method should remove the value at the given index. The removed value should be returned by the method. You must check that the given index is valid. If it is invalid error handling using exceptions should be performed. Your implementation must be compatible with the store-method above. There must be no “holes” in the array and the attribute number must be updated properly.

The second method should sort the array in ascending order. One way of doing this is selection sort:

1) set number of values to n, ie all values
2) find the max value among these values
3) exchange n'th value and the max value
4) decrease n by one and if n>1 go back to point 2
9. **Fortran**

Write a subroutine declared as

```fortran
subroutine findtrailing( a, b, c )
```

Here, `a` is an integer variable, (grade 5 see below), with a value $0 - 99$, and `b` is an array (possibly long) containing positive integers, all of which having values that are at least 1000. The parameter `c` is a pointer, that can point to an array of integers. The subroutine should go through the array `b` and check which elements have a value that (in the decimal system) end with the same two digits as `a`. The subscripts of these elements should be returned in an array pointed to by `c`. This array has to be allocated by the subroutine.

For 4 points on this problem (aiming for grade 4), the formulation of the problem is as above. For 5 points (aiming for grade 5), `a` should be a (short) array of values between 0 and 99. The resulting `c` should point to a two-dimensional array of subscripts, Row $j$ in this array should contain the subscripts of `b`-elements that end in the same digits as $a(j)$. Note that this two-dimensional array should be a normal array, i.e., all rows should have the same length, although not all of them might be filled with nonzero elements.

10. **Fortran**

A data file, named `secdata.dat`, is structured so that it contains data values in *sections* as shown in the following example:

```plaintext
'PART1' 4
.
  4 lines each containing one real number
.
'SECTDATA' 7
.
  7 lines each containing one real number
.
'END’ 0
```

Note that the file above is just an *example* of how the file could look like, there can be an arbitrary number of sections. Each section is headed by a line containing a string, which is the name of the section followed by an integer giving the number of lines in the section. Each line in the section contains one real data value. After the last section comes a line with the name ’END’ and an arbitrary integer value.

Write a main program that reads the file, and for each section of data finds the sum of the values contained in that section. The program should write its output to a file `secdata.out`. For each section the name of the section and the sum of the
values should be written on one line of the output file.

For 4 points, (aiming for grade 4), the problem can be solved according to the specifications above. For 5 points (aiming for grade 5), the program should handle the situation that the file does not contain a trailing line with 'END' xx, but instead end-of-file follows directly after the last section.