Exercise in C++ Programming for CE
ExC++PCE
Winter Term 2014/2015

Assignment sheet C

Assignments that are marked with *StudOn submission* are **mandatory** and must be submitted via StudOn in time – please see there for deadlines.

1 Expressions

StudOn submission

Given the following statement, in which order are the contained expressions evaluated and which conversions take place? Consider operator precedence and associativity!

```
float a, b, c;
a = b + c * 2.;
```

(a) Expression 1: _____ Conversions: _____

(b) Expression 2: _____ Conversions: _____

(c) Expression 3: _____ Conversions: _____

2 Statements (`C2_stmnt.cpp`)

Each of the following programs contains a common programming error. Identify and correct each error.

```
(a)   switch( ival ) {
          case 1:
              int ix = get_value();
              ivec[ ix ] = ival;
              break;
          default:
              ix = ivec.size()-1;
              ivec[ ix ] = ival;
      }
```

(b)
```
switch( ival ) {
   case 'a': aCnt++;
   case 'e': eCnt++;
   default: iouCnt++;
}
```

(c)
```
switch( ival ) {
   case 1, 3, 5, 7, 9:
      oddcnt++;
      break;
   case 2, 4, 6, 8, 10:
      evencnt++;
      break;
}
```

(d)
```
int ival=512;
int jval=1024;
int kval=4096;
int bufsize;
// ...
switch( swt ) {
   case ival:
      bufsize = ival*sizeof(int);
      break;
   case jval:
      bufsize = jval*sizeof(int);
      break;
   case kval:
      bufsize = kval*sizeof(int);
      break;
}
```

3 Functions

StudOn submission

Given the following declarations, determine which function calls are correct and which are invalid.

```
int calc( int, int );
double calc( double, double=2.4 );
int count( const string&, char )
int sum ( vector<int>::iterator, vector<int>::iterator, int );
vector<int> vec;
```

(a)	`calc( 23.4, 55.1 );`	☐ correct	☐ invalid
(b)	`count( "abcda", 'a');`	☐ correct	☐ invalid
(c)	`calc( 66 );`	☐ correct	☐ invalid
(d)	`sum( vec.begin(), vec.end(), 3.8 );`	☐ correct	☐ invalid
(e)	`calc( 27.3, 12 );`	☐ correct	☐ invalid

4 Fibonacci Function (`C4_fib.cpp`)

Write a function to produce the Fibonacci sequence. Use this function to fill a `vector<string>` with the first ten Fibonacci numbers. Sort the vector lexicographically and print the resulting sequence to the screen.

5 Memory Function (`C5_memory.cpp`)

Write a function that returns 0 when it is called for the first time and then generates sequential numbers each time it is called again (i.e., 1, 2, 3, ...). Use this function in a suitable `main` function to demonstrate its correctness.

6 Tiny Program

Indicate whether the following program is correct. If so, explain what it does; if not, make it legal and then explain it:

```
int& get( int* array, int index ) { return array[index]; }
int main() {
   int ia[10];
   for( int i = 0; i != 10; ++i )
      get( ia, i ) = 0;
}
```

7 Limited Function

The following function, although legal, is less useful than it could be. Identify and correct the limitation on this program!

```
bool test( string& s ) { return s.empty(); }
```

8 Vector Addition (`C8_vecadd.cpp`)

The following function implements the addition of two dense vectors. Examine the function and comment on the correctness of the function.

```
Vector& addVectors( const Vector& v1, const Vector& v2 )
{
   // Throwing an exception in case the sizes of the two
   // vectors don't match
   if( v1.size() != v2.size() )
      throw std::invalid_argument( "Vector sizes don't match" );

   Vector tmp( v1.size() );
   for( size_t i = 0; i < v1.size(); ++i )
      tmp[i] = v1[i] + v2[i];

   return tmp;
}
```

9 Function Overloading

Explain the effect of the second definition in each one of the following sets of definitions. Indicate whether the second definition is OK, whether it results in ambiguous function calls, or whether the definition results in a compilation error:

```
(a)  int calc( int, int ) { ... }
     int calc( const int&, const int& ) { ... }


(b)  int calc( char*, char* ) { ... }
     int calc( const char*, const char* ) { ... }


(c)  int calc( char*, char* ) { ... }
     int calc( char* const, char* const ) { ... }
```