

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;

namespace Lab2ServerCS
{
    /*
    * This random number generator originally appeared in "Toward a Universal
    * Random Number Generator" by George Marsaglia and Arif Zaman.
    * Florida State University Report: FSU-SCRI-87-50 (1987)
    *
    * It was later modified by F. James and published in "A Review of Pseudo-
    * random Number Generators"
    *
    * THIS IS THE BEST KNOWN RANDOM NUMBER GENERATOR AVAILABLE.
    * (However, a newly discovered technique can yield
    * a period of 10^600. But that is still in the development stage.)
    *
    * It passes ALL of the tests for random number generators and has a period
    * of 2^144, is completely portable (gives bit identical results on all
    * machines with at least 24-bit mantissas in the floating point
    * representation).
    *
    * The algorithm is a combination of a Fibonacci sequence (with lags of 97
    * and 33, and operation "subtraction plus one, modulo one") and an
    * "arithmetic sequence" (using subtraction).
    * =====
    * This C language version was written by Jim Butler, and was based on a
    * FORTRAN program posted by David LaSalle of Florida State University.
    * -----
    * Converted to C++ (MSVC6) by M. Boldin, Niagara College Canada
    * Ported to C# (MSVS2K8), 2009.10.23
    */
    class RandomException : Exception
    {
        public RandomException(string reason)
            : base(reason)
        {
        }
    }

    class Random
    {
        private const int MAX_IJ = 31328 ;
        private const int MAX_KL = 30081 ;

        public static void
        frandinit( )
        {
            int ij, kl ;
            DateTime now = DateTime.Now ;

            ij = (int) ( now.Ticks % (long) MAX_IJ ) + 1 ;
            kl = (int) ( now.Ticks % (long) MAX_KL ) + 1 ;

            try
            {
                rmarin( ij, kl ) ;
            }
            catch ( RandomException ex )
            {
                Console.WriteLine( ex ) ;
            }
        }

        public static double
        frand( )
    }
}
```

```
{
    double [] r = new double [ 2 ] ;

    try
    {
        ranmar(r, 1);
    }
    catch (RandomException ex)
    {
        Console.WriteLine(ex);
        r[0] = r[1] = -1.0;
    }

    return ( r[ 1 ] ) ;
}

public static void
main( )
{
    double [] temp = new double [ 101 ] ;
    int ij, kl, len, i ;

    /* random seeds for the test case: */
    ij = 1802 ;
    kl = 9373 ;

    try
    {
        /* do the initialization */
        rmarin(ij, kl);

        /* generate 20,000 random numbers */
        len = 100;
        for (i = 1; i <= 200; i++)
        {
            ranmar(temp, len);
        }

        /*
         * If the random number generator is working
         * properly, the next six random numbers should be:
         *
         * 6533892.0  14220222.0  7275067.0
         * 6172232.0  8354498.0  10633180.0
         */

        len = 6;
        ranmar(temp, len);
    }
    catch (RandomException ex)
    {
        Console.WriteLine(ex);
        return;
    }

    Console.WriteLine( "If the random number generator is working\n" +
        "properly, the next six random numbers should be:\n" );
    Console.WriteLine( "\t{0, 12: #.0}", 6533892.0 );
    Console.WriteLine( "\t{0, 12: #.0}", 14220222.0 );
    Console.WriteLine( "\t{0, 12: #.0}", 7275067.0 );
    Console.WriteLine( "\t{0, 12: #.0}", 6172232.0 );
    Console.WriteLine( "\t{0, 12: #.0}", 8354498.0 );
    Console.WriteLine( "\t{0, 12: #.0}\n", 10633180.0 );

    for ( i = 1 ; i <= 6 ; i++ )
    {
        Console.WriteLine( "\t{0, 12: #.0}", 4096.0 * 4096.0 * temp[ i ] );
    }
}
```

```

}

private static double [] u ;
private static double c, cd, cm ;
private static int i97, j97 ;
private static bool seeded_b = false ;

public static void
rmarin( int ij, int kl )
{
    /*
    * This is the initialization routine for the
    * random number generator RANMAR()
    *
    * NOTE: The seed variables can have values between:
    * 0 <= IJ <= 31328
    * 0 <= KL <= 30081
    *
    * The random number sequences created by these two seeds
    * are of sufficient length to complete an entire calculation
    * with. For example, if several different groups are
    * working on different parts of the same calculation,
    * each group could be assigned its own IJ seed. This would
    * leave each group with 30000 choices for the second seed. That
    * is to say, this random number generator can create 900
    * million different subsequences -- with each subsequence
    * having a length of approximately 10^30.
    *
    * Use IJ = 1802 & KL = 9373 to test the random number
    * generator. The subroutine RANMAR should be used to
    * generate 20000 random numbers. Then display the next
    * six random numbers generated multiplied by 4096 * 4096
    * If the random number generator is working properly,
    * the random numbers should be:
    *
    * 6533892.0 14220222.0 7275067.0
    * 6172232.0 8354498.0 10633180.0
    */
    int i, j, k, l, ii, jj, m ;
    double s, t ;

    if ( ( ij < 0 ) || ( ij > MAX_IJ ) || ( kl < 0 ) || ( kl > MAX_KL ) )
    {
        string exmsg = string.Format( "The first random number seed must have a " +
            "value between 0 and {0}.\n" +
            "The second seed must have a value between " +
            "0 and {1}.",
            MAX_IJ, MAX_KL ) ;
        throw new RandomException( exmsg ) ;
    }

    if ( u == null )
    {
        u = new double [ 98 ] ;
    }

    i = ( ij / 177 ) % 177 + 2 ;
    j = ij % 177 + 2 ;
    k = ( kl / 169 ) % 178 + 1 ;
    l = kl % 169 ;

    for ( ii = 1 ; ii <= 97 ; ii++ )
    {
        s = 0.0 ;
        t = 0.5 ;

        for ( jj = 1 ; jj <= 24 ; jj++ )

```

```

    {
        m = ( ( ( i * j ) % 179 ) * k ) % 179 ;
        i = j ;
        j = k ;
        k = m ;
        l = ( 53 * l + 1 ) % 169 ;

        if ( ( ( l * m ) % 64 ) >= 32 )
        {
            s += t ;
        }

        t *= 0.5 ;
    }

    u[ ii ] = s ;
}

c = 362436.0 / 16777216.0 ;
cd = 7654321.0 / 16777216.0 ;
cm = 16777213.0 / 16777216.0 ;

i97 = 97 ;
j97 = 33 ;

seeded_b = true ;
}

public static void
ranmar( double [] rvec, int len)
    /* len random numbers are placed in rvec[1..len] */
{
    /*
    * This is the random number generator proposed by
    * George Marsaglia in Florida State University Report:
    * FSU-SCRI-87-50
    * It was slightly modified by F. James to produce an
    * array of pseudorandom numbers.
    */

    int ivec ;
    double uni ;

    if ( ! seeded_b )
    {
        throw new RandomException(
            "Call the init routine rmarin() before " +
            "calling ranmar()."
        ) ;
    }

    for ( ivec = 1 ; ivec <= len ; ivec++ )
    {
        uni = u[ i97 ] - u[ j97 ] ;

        if ( uni < 0.0 )
        {
            uni += 1.0 ;
        }

        u[ i97 ] = uni ;
        i97-- ;

        if ( i97 == 0 )
        {
            i97 = 97 ;
        }
        j97-- ;
    }
}

```

```
        if ( j97 == 0 )
        {
            j97 = 97 ;
        }

        c -= cd ;
        if ( c < 0.0 )
        {
            c += cm ;
        }

        uni -= c ;
        if ( uni < 0.0 )
        {
            uni += 1.0 ;
        }

        rvec[ ivec ] = uni ;
    }
}
}
```