STAT - 611 ASSIGNMENT 2

Due Thursday Oct 8, 2015 in class

Remember (from syllabus): Solutions must be neatly word-processed (using Latex or a word processor) and stapled. Please annotate your work with brief, clear sentences explaining your approach and interpreting your results (you are not expected to write full-blown data analysis reports however). For assignments involving mathematical manipulations, students can write answers by hand provided penmanship is neat; illegible answers will be marked as incorrect.

*All these questions require you writing small C/C++ programs. Please write a report with your code, an explanation of what you've done, and your code. Additionally attach the files containing the code you've written to OnCourse with the titles

- `yourlastname_hw1_q1.cpp`
- `yourlastname_hw1_q2.cpp`

1. A method for determining the first m digits $0.b_1b_2b_3...b_m$ of the binary representation of a real number $x \in [0, 1)$ is the following algorithm:

   ```
   c <- x
   for j from 1 to m do
     b[j] <- floor(2c)
     c <- 2c - b[j]
   ```

   where `floor(x)` is the greatest integer less than or equal to the real number x.

   (a) Explain how this algorithm works.

   (b) Write a C function that implements this algorithm. The prototype of your function should be

   ```
   void get_binary(double x, int* b, int m);
   ```

   where x is the real number in double precision floating point representation, b is the starting address of an array of integers, and m is the number of digits we want to obtain. You will need to include the header file "math.h"; also include the "stdio.h" header:

   ```
   #include<stdio.h>
   #include<math.h>
   ```

   Test your function with the following "main" function:

   ```
   int main(){
     int b[30];
     get_binary(0.1, b, 30);
     for(int i = 0; i < 30; i++){
       printf("%d", b[i]);
     }
     return 0;
   }
   ```

2. Write a C implementation of Kahan's summation algorithm. Your task is to write a function that takes an array of double precision floating point numbers and returns its sum. The prototype of your function should be:

   ```
   double kahan_sum(double* a, int length);
   ```

For comparison also write a function that implements the naive sum of all the elements of the array in order (i.e. a[0] + a[1] + ... + a[length - 1]). The prototype of this function should be:

```
double naive_sum(double* a, int length);
```

Test your functions with the following code:

```
int main (){
  double table[10000];
  table[0] = 1.0;
  for (int i = 1; i < 10000; i++){
    table[i] = 1e-16;
  }
  printf("naive = %1.14e\n", naive_sum(table, 10000));
  printf("kahan = %1.14e\n", kahan_sum(table, 10000));
  return(0);
}
```

and discuss your results.