We wrote a program to compute $N!$. We will modify it a bit to do something that looks a bit strange, namely to compute the following sum:

$$1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \frac{x^3}{3!} + \cdots + \frac{x^N}{N!}$$

Warning: Factorials grow very fast, so the problem we encountered with overflowing for the arithmetic series could occur here quite easily. One way to make it less likely to happen is to use double precision numbers to store our factorials instead of integers. More on this later. You can write the code from scratch or else make a copy of your factorial program and modify. If you copy and modify, make sure you declare "prod" to be double.

```
/* this is a program to do a weird sum*/
#include <stdio.h>
#include <math.h>
int main(void)
{
double prod=1, sum=1, x;
int j, N;
printf("Enter N and x");
printf("\n");
scanf("%i %lf",&N,&x);
for (j=1; j<N; j=j+1)
{
prod=prod*j;
sum=sum+pow(x,j)/prod;
printf("\n   %i",j);
printf("  ");
printf("%20.10lf",sum);
}
printf("\n");
return 0;
}
```

[1] You will need -lm when you compile because of the 'pow' function.

[2] Run your program with $N = 10$ and $x = 1$. Do you recognize the number you are getting? What about $N = 10$ and $x = 0$? What about $N = 10$ and $x = 2$? *Query to the class: Has everyone seen the exponential function?*

[3] So here's the amazing fact: If N is large enough,

$$e^x = 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \frac{x^3}{3!} + \cdots + \frac{x^N}{N!}$$

I say this is amazing because we think of the exponential as this bizarre function, but, actually, it is just a polynomial! More precisely, it is a polynomial of very high (infinite) order.

[4] You'll spend a lot of time on this if you take a calculus course. The equation above is called the 'Taylor expansion' of the exponential.