Homework 2  
Due September 23rd

1. Problem 3.2 from book

```c
float xmin = 1.0;
float lastxmin;

while ( xmin > 0 ) {
    lastxmin = xmin;
    xmin = xmin/2;
}

xmin = lastxmin;
printf ("The smallest number is %g\n", xmin);
```

On my PowerPC Mac, for single-precision floats I get $1.4013 \times 10^{-45}$ which is equal to $2^{-150}$, and for double-precision I get $4.94066 \times 10^{-324}$ which is equal $2^{-1074}$. I get the same answers on Fester, a Pentium-IV machine.

2. Problem 3.8 from book
a) $y = x^3 - 5x^2 + 6x + 0.55$
   
   $$
   = (2.73)^3 - 5(2.73)^2 + 6(2.73) + 0.55
   = 20.3 - 37.2 + 16.3 + 0.55
   = -0.05
   $$
   
   True value is 0.011917, so the error is $(-0.05 - 0.011917)/0.011917 = 519\%$

b) $y = [(x - 5)x + 6]x + 0.55$
   
   $$
   = [(2.73 - 5)2.73 + 6]2.73 + 0.55
   = [(-2.27)2.73 + 6]2.73 + 0.55
   = [-6.19 + 6]2.73 + 0.55
   = [-0.19]2.73 + 0.55
   = -.518 + 0.55
   = 0.032
   $$
   
   Error is $(0.032 - 0.011917)/0.011917 = 168\%$.
   The error is significantly reduced by rearranging the equation so that there are no high order terms
### 3. Problem 3.10 from book

```c
{ 
    double pi = 3.14159265359;
    double es = 0.5e-08;
    double fact = 1;
    int j = 1;
    double x = 2*pi;
    double term = 1;
    double cosx = term;

    printf("j=%d   cos(x)= %0.10f\n", j, cosx);

    for ( j=2; j<100; j++ ) {
        double ea, newcosx;

        fact = fact*(2*j-3)*(2*j-2);
        term = (-term) * x * x;

        newcosx = cosx + term/fact;
        ea = (newcosx-cosx)/newcosx;
        cosx = newcosx;

        printf("j=%d   cos(x)= %0.10f, ea=%.10f%%\n", j, cosx, ea*100);
        if ( (ea < 0 && (-ea) < es) || (ea > 0 && ea < es) )
            break;
    }
}
```

| j=1 | cos(x)= 1.0000000000 |
| j=2 | cos(x)= -18.7392088022, ea=105.3364045972% |
| j=3 | cos(x)= 46.2001852205, ea=140.5608954006% |
| j=4 | cos(x)= -39.2566319862, ea=217.6875928549% |
| j=5 | cos(x)= 20.9880093857, ea=287.0431409898% |
| j=6 | cos(x)= -5.4382473977, ea=485.9333320241% |
| j=7 | cos(x)= 2.4652889736, ea=320.5926954570% |
| j=8 | cos(x)= -0.7508982625, ea=228.3119826805% |
| j=9 | cos(x)= 1.0329042310, ea=27.3022377096% |
| j=10 | cos(x)= 0.9965213898, ea=3.6509844659% |
| j=11 | cos(x)= 1.0000013329, ea=0.3778695967% |
| j=12 | cos(x)= 0.9999782330, ea=0.0322998098% |
| j=13 | cos(x)= 1.0000001332, ea=0.0023099926% |
| j=14 | cos(x)= 0.9999999299, ea=0.0001403000% |
| j=15 | cos(x)= 1.0000000032, ea=0.0000073265% |
| j=16 | cos(x)= 0.9999999999, ea=0.0000003325% |

### 4. Problem 4.6 from book

\[ f(x) = 25x^3 - 6x^2 + 7x - 88 \]
\[ f'(x) = 75x^2 - 12x + 7 \]

True value of \( f'(2) \) is 283.
Forward-difference

\[ f'(2) = \frac{f(2 + .25) - f(2)}{.25} = \frac{182.14 - 102}{.25} = 320.5625 \]

\[ \epsilon_i = \left| \frac{283 - 320.5625}{283} \right| = 13.3\% \]

Backward-difference

\[ f'(2) = \frac{f(2) - f(2-.25)}{.25} = \frac{39.859 - 102}{.25} = 248.5625 \]

\[ \epsilon_i = \left| \frac{283 - 248.5625}{283} \right| = 12.2\% \]

Centered-difference

\[ f'(2) = \frac{f(2 + .25) - f(2 -.25)}{2(.25)} = \frac{182.14 - 39.859}{2(.25)} = 284.5625 \]

\[ \epsilon_i = \left| \frac{283 - 284.5625}{283} \right| = 0.55\% \]

The expected error for the forward and backward difference methods is the Taylor Series remainder: 

\[ f''(2) \left(0.25\right)\frac{1}{2!} = \frac{150(2) - 12(0.25)}{2} = 36. \]

The actual absolute errors are 80.14 and 62.141. The expected error for the centered-difference method is

\[ -\frac{f^{(3)}(2)}{3!}(0.25)^2 = -\frac{150}{6}(0.25)^2 = -1.5625 \]

which matches the actual absolute error.
5. Problem 4.10 from book

\[
\Delta H(\tilde{T}) = \left| \frac{\partial H}{\partial \tilde{T}} \right| \Delta \tilde{T} \\
= 4 Ae \sigma T^3 \Delta \tilde{T} \\
= \left| 4(0.15)(0.90)(5.67e - 8)(650)^3 \right| (25) \\
= 210.21
\]

Exact error

\[
\Delta H_{\text{exact}} = \frac{H(675) - H(625)}{2} \\
= \frac{(0.15)(0.90)(5.67e - 8)(675)^4 - (0.15)(0.90)(5.67e - 8)(625)^4}{2} \\
= \frac{1589 - 1168}{2} \\
= 210.52
\]

The estimated error is very close to the exact error.

With \(\Delta T=50\):

\[
\Delta H(\tilde{T}) = \left| \frac{\partial H}{\partial \tilde{T}} \right| \Delta \tilde{T} \\
= 4 Ae \sigma T^3 \Delta \tilde{T} \\
= \left| 4(0.15)(0.90)(5.67e - 8)(650)^3 \right| (50) \\
= 420.42
\]

Exact error

\[
\Delta H_{\text{exact}} = \frac{H(700) - H(600)}{2} \\
= \frac{(0.15)(0.90)(5.67e - 8)(700)^4 - (0.15)(0.90)(5.67e - 8)(600)^4}{2} \\
= \frac{1838 - 992}{2} \\
= 422.9
\]

Again, the estimated error is very close to the exact error.

The estimated errors are very close to the exact error because the function is nearly linear around \(T=650\).
6. The following is an algorithm to calculate \( e^x \) using a Taylor Series expansion.

\[
\begin{align*}
e_x & = 1 \\
\text{for } i=1 \text{ to } n \text{ do } \\
& \quad \text{nfact} = 1 \\
& \quad \text{xpower} = 1 \\
& \quad \text{for } j = 1 \text{ to } i \text{ do } \\
& \quad \quad \text{nfact} = \text{nfact} \times j \\
& \quad \quad \text{xpower} = \text{xpower} \times x \\
& \quad \text{endfor} \\
& \quad e_x = e_x + \text{xpower}/\text{nfact} \\
\text{endfor}
\end{align*}
\]

a. What is the order of growth of this algorithm relative to \( n \)?

The order of growth of this algorithm is determined by the number of operations in the algorithm. The number of multiplications \( = \sum_{i=1}^{n} 2i = n(n + 1) \), and the number of divisions is \( n \). Put them together, the overall growth is therefore \( O(n^2) \).

b. Can this algorithm be rewritten to have a lower order of growth? If so, show the new algorithm and compute its order of growth.

\[
\begin{align*}
nfact & = 1 \\
xpower & = 1 \\
e_x & = 1 \\
\text{for } i=1 \text{ to } n \text{ do } \\
& \quad \text{nfact} = \text{nfact} \times i \\
& \quad \text{xpower} = \text{xpower} \times x \\
& \quad e_x = e_x + \text{xpower}/\text{nfact} \\
\text{endfor}
\end{align*}
\]

The number of multiplications \( = 2n \), and the number of divisions is \( n \). Put them together, the overall growth is therefore \( O(n) \).