Hints for assignment 1.

For people who are new or rusty at programming.

Below is a list of steps that might help someone overcome the strictly computer related barriers that face a person writing a first program. The main point is to learn every new thing in the simplest possible context before incorporating it into a larger code. It is written with C++ in mind. Probably a person learning programming for the first time should learn C++. A novice programmer has a C++ book at her/his side.

(A) Follow the instructions in the book to make `"Hello, world."` appear on the screen.

(B) Write a `for` loop that adds up the numbers from 1 to n and prints the sum.

(C) Write a program with a variable `int error_code`. Set `error_code` 0 and 1. Write an `if` test that prints the message "`have a non zero error code = `" and then the value of `error_code` if `error_code` is not zero.

(D) Write a program with a procedure `int panel_method(float *f, int n)` that sets `f` to be $2n$ and returns an error code (a non zero value) if $n \leq 0$ but otherwise returns zero. Test this procedure by calling it with various n values and printing the results:

```
int return_code;
float f;
  .
  .
  .
return_code = panel_method( &f, n);
if ( return_code ) {  \\ Print an error message if the return code is
                      \\ not zero.
   cout << "panel_method returned error code: " << error_code
        << "  with n = " << n << endl; }
Print n and f  \\ Otherwise, print the results.
  .
  .
  .
```

(E) Change the insides of `panel_method` to be a panel method. Call it with $n = 1000$ and $x = .2$ and compare the result to a Taylor approximation. Repeat with $x = .1$. With $x = .2$, make runs with both the trapezoidal rule and Simpson's rule to make sure it is easy to make this change in the code. Make sure there are comments in the code that tell someone exactly what to change to put in a different panel method.

(F) Write a procedure `int fake(float* I, int n)` that returns a non zero error code if $n \le 0$ and sets $I = 1/\sqrt{2} + 1/n^2 + 2/n^4 + 5/n^6$. Define a global constant, `N_0` to be 20 and compute

$$R(N_0) = \frac{I(2N_0) - I(N_0)}{I(4N_0) - I(2N_0)} \quad .$$

This should be around 4. Compute $R(2N_0)$ and $R(4N_0)$ to see whether the value is getting closer to 4. It might be convenient to define a procedure that computes $R(N)$ and call it several times. Remove the $1/n^2$ term from I and repeat. You should see the higher order of accuracy.

(G) Do Step 3 of the assignment by replacing `fake` by `panel_mathod`.

(H) Write a procedure that takes as argument ϵ and x. It should call `panel_method` with n values starting from N_0 and doubling until $|f_{2n}(x) - f_n(x)| \le \epsilon$ or $n \ge N_{MAX}$ with $N_{MAX} = 10^6$. Return a non zero error code in the latter case. Here, $f_n(x)$ is the value of $f(x)$ estimated by the panel integration method with n panels. The procedure should supply the most accurate f value it has computed.

(I) Write a program that opens a file and writes "`Hello, file.`" into it. On a UNIX system the following hints should work:

```
#include <iostream.h>
#include <fstream.h>
   .
   .
   .
int main() {
   .
   .

      \\ Open a file called plot.mat for output.

    ofstream output_file( "plot.mat", ios::out);

       \\ Print an error message if the file didn´t open properly.
      if ( !output_file ) {
         cout << "Could not open the output file plot.mat" << endl:}

    output_file << "Hello, file." << endl;
        .
        .
```

(J) Now write into the output file:

```
1    1
2    4
3    9
4   16
5   25
```

Read the file into Matlab, plot it and see whether it looks like a parabola.

(K) Do Step 4 of the assignment. Make a loop that computes $f(x)$ and prints lines into the plot file that contain x, $f(x)$, and n. This n is the n needed to achieve the accuracy requested for that x value.