CS 007A, Spring 2013 – Midterm 1, Chapters 1—5. Name:_____
Write all responses on separate paper. You may not use a compiler, though a hand-held calculator is ok.

1. Which of the following declarations are illegal, and why? Be sure to state your reasons.
   a. `int float;` ANS: `float` is a key word, can't be use for a variable identifier.
   b. `float F3.7;` ANS: can't use punctuation like `"."` In a variable name.
   c. `bool true?;` ANS: can't use punctuation like `"?"` In a variable name.
   d. `unsigned next-number;` ANS: can't use operations like `"-"` In a variable name.
2. Each of the following program fragments has at least one error. Locate the error(s), classify each error as either a syntax error, or a logical error and suggest how it could be corrected (assume that the usual preprocessor commands `#include <iostream>; using namespace std;` are present)
   a.
```
#include <iostream>;
using namespace std;
int main() {
    int n17 == 17;
    cout << n/2 << endl;
}
```
   ANS: The semicolon at the end of "`#include <iostream>;`" is a syntax error.
   Also, you don't use the comparison operator "==" to assign a value. That will cause a syntax error.
   b.
```
#include <iostream>
using namespace std;
int main() {
    float n /= 10.1;
    cout << n*10.1:
}
```
   ANS: You can't divide a variable that hasn't been initialized yet. Also, : where there ought to be a ;.
   c.
```
int i;
double pi;
pi = 1.0;
for (i=2; i<n, i=++i)
{  if(i%2 = 1)
      pi = pi - (1.0 / ((2.0*i)+1));
   else
      pi = pi + (1.0 / ((2.0*i)+1));
}
pi = pi * 4.0;
```
   SOLN: The assignment `i=++i` is not allowed. Also, the `i` in the for loop was never declared. There is a comma separating the fields in the for structure which should be a semicolon, and there is an assignment operator in the if conditional which should be a comparison operator.
3. What is output to the console by each of the following statement?
   a. `cout << 1 + 2 * 3 / 4 << endl;`
      `SOLN 1 + 2 * 3 / 4 = 1 + 6 / 4 = 1 + 1 = 2,` So the output is `2.`
   b. `cout << 1 * 2 + 3 / 4 + 2 << endl;`
      `SOLN: 1 * 2 + 3 / 4 + 2 = 2 + 0 + 2 = 4,` So the output is `4.`
   c. `cout << 2 * (2 + 3) / 4 << endl;`
      `SOLN: 2 * (2 + 3) / 4 = 2 * 5 / 4 = 10 / 4 = 2,` So the output is `2.`
   d. `cout << 1 / 2 * 2 + 3 << endl;`
      `SOLN: 1 / 2 * 2 + 3 = 0 * 2 + 3 = 3,` the output is `3.`
   e. `cout << 1. / 2 * 2 + 3 << endl;`
      `SOLN: 1. / 2 * 2 + 3 = 0.5 * 2 + 3 = 1 + 3 = 4,` So the output is `4.`

4. Determine what is print to the console by the following code fragment, assuming the declarations `int a=2, b=3, c=4, d=5; double A[4] = {1.1, 2.2, 3.3, 4.4};`
If the code is badly formed, explain why.

a. `cout << b/(a+d%c);`
SOLN: `3/(2+5%4)= 3/(2+1) = 3/3 = 1,` So the output is `1.`

b. `if(a-b>0 && 1/(b-5)==0) cout << "ok";`
SOLN: Plugging into the conditional, `a-b>0 && 1/(b-5)==0` we have `2-3>0 &&` … but we don't have to evaluate the second condition, since the first false already makes the conjunction false. Thus there is no output.

c. `if(a = 3) cout << d/A[0]; else cout << a/A[0];`
SOLN: This looks like a logic error, and it almost certainly is, but it's not a syntax error since 3 is successfully assigned to `a`, the conditional is evaluated as `true and d/A[0] = 5/1.1 = 4.54545` (note that d is promoted to a double because it's an int divided by a double. So the output is `4.54545`

d. `do { cout << A[c--]; } while(c>0);`
SOLN: Well, there was supposed to be a curly brace before the `while(c>0)` and its omission causes a syntax error. Supposing the curly brace were there, there is a logic error, since the first value to be printed would be `A[4],` and no memory has been allocated for this. Thus when you run code with this expression you'll get
`-9.25596e+0614.43.32.2,`
or something similar, where the first number, `-9.25596e+061`, is just memory garbage.

e. `while(--a>=0) cout << A[a];`
SOLN: Here `a` is decremented before being compared with 0, so the output will be `A[1]` followed by `A[0]:`
2.21.1

5. Consider the following complete program.

```
#include <iostream>
using namespace std;
int size = 10;
int main() {
    int a, b;
    cout << "\nEnter two prime numbers: ";
    cin >> a >> b;
    for(int k = 1; k < size; ++k)|
    {   if(k%a==0) cout << a;
        else if(k%b==0) cout << b;
        else cout << a+b;
        cout << endl;
    }
}
```

a. What is the scope of the variable "size"?
SOLN: This has global scope.

b. What is the scope of the variable "a"?
SOLN: This has the scope of `main()`.

c. What is the scope of the variable "k"?
SOLN: This has the scope of the `for` loop.

d. What is the output on the console of this program if the user enters "2 3"?
SOLN: The
Enter two prime numbers: 2 3
5
2
3
2
5
2
5
2
3

6. Implement the following function which uses Horner's method to evaluate the polynomial

$$2x^5 - 7x^4 + 6x^3 + 9x^2 + 8x + 5$$

```
double p(double x);
// Returns 2x^5 - 7x^4 + 6x^3 + 9x^2 + 8x - 5.
// EXAMPLE: p(4.1) returns 931.70732
SOLN:  Here's one complete implementation:
```

```
#include <iostream>
using namespace std;

double p(double x)
// Returns 2x^5 - 7x^4 + 6x^3 + 9x^2 + 8x - 5
// using horner's form: ((((2*x-7)*x + 6)*x + 9)*x +8)*x - 5
// EXAMPLE: p(4.1) returns 931.70732
{
   double coeff[6]={-7,6,9,8,-5};
   double value = 2;
   for(int i = 0; i < 5; ++i)
   {
        value *= x;
        value += coeff[i];
   }
   return value;
}

int main()
{
   cout << "\np(4.1) = " <<  p(4.1) << endl;
}

Output:
p(4.1) = 931.707
```

7. Implement the following `compare()` function:

```
int compare(float x[], float y[], int n);
// Returns 0 if x[i] == y[i]
// or returns -1 if x[i] < y[i]
// or returns 1 if x[i] > y[i] for 0 <= i < n.
// PRECONDITION: x and y both have at least n elements.
```

SOLN: This was not a good problem statement because arrays of values are not well ordered the way, say,

the real numbers are. We can say whether or not one array is equal to another or not (each and every component value is the same), but to say one is less than or greater to another is not quite so clear. The best interpretation of this is to impose a place value system on the array, so that floats are weighted more heavily as you approach one end or another. Then it's essentially like ordering integers or putting words in alphabetical order.

```
int compare(float x[], float y[], int n)
// Returns 0 if x[i] == y[i]
// or returns -1 if x[i] < y[i]
// or returns 1 if x[i] > y[i] for 0 <= i < n.
{
   int i = 0;
   for(; i < n && x[i] == y[i]; ++i);
   if(i == n) return 0;
   else return x[i] < y[i] ? -1 : 1;
}


int main()
{
   //cout << "\np(4.1) = " <<  p(4.1) << endl;
   float x[size] = {1.1, 2.2, 3.3};
   float y[size] = {1.1, 2.2, 3.3};
   cout << "\nIf they're equal, then compare(x,y,size) = "
        << compare(x,y,size) << endl;
   y[2] = 4.4;
   cout << "\nIf x < y, then compare(x,y,size) = "
        << compare(x,y,size) << endl;
   y[2] = 2.2;
   cout << "\nIf x > y, then compare(x,y,size) = "
        << compare(x,y,size) << endl;
}
Output:

If they're equal, then compare(x,y,size) = 0

If x < y, then compare(x,y,size) = -1

If x > y, then compare(x,y,size) = 1
```

8. Write a C++ program to implement the flow chart to the right. Note that the structure of the algorithm is suited to two for loops followed by a while loop with a for loop and a while loop nested in it.

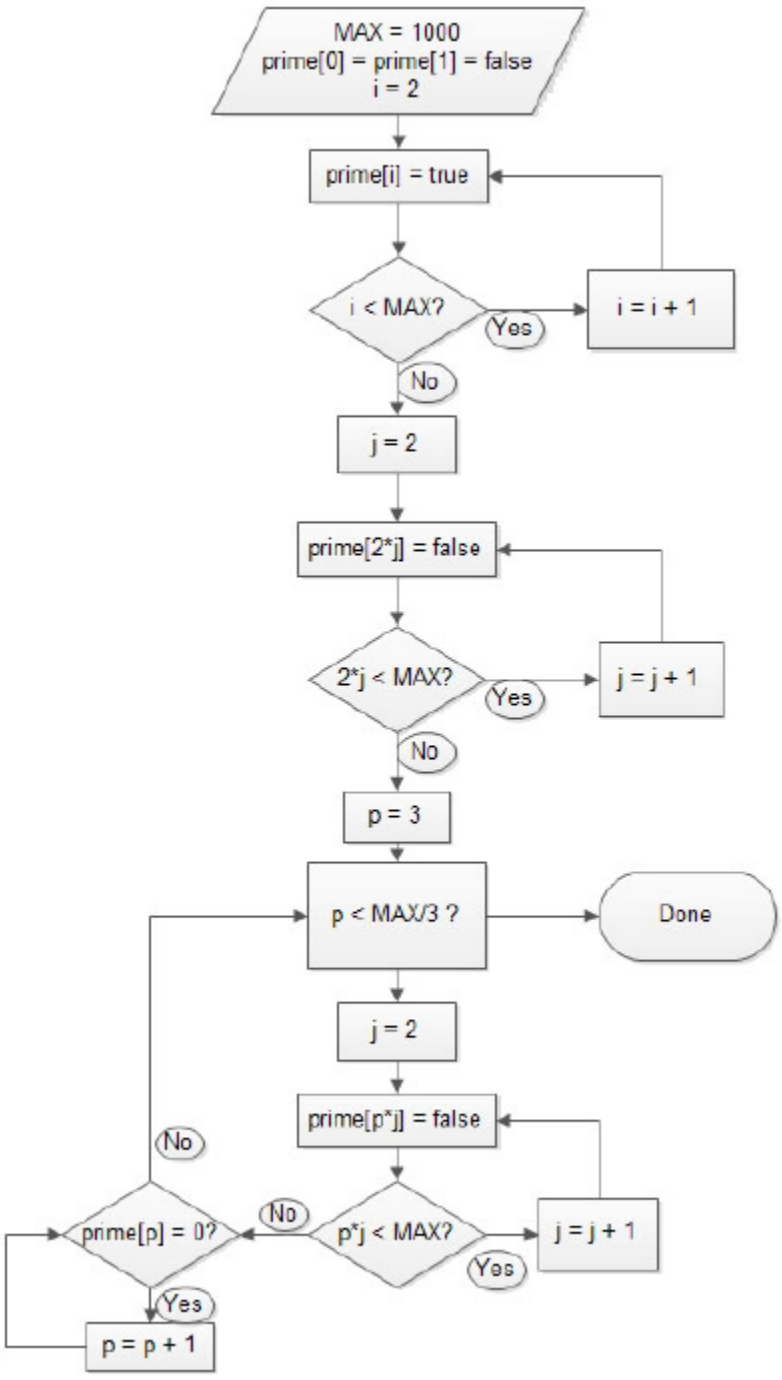
```
const int MAX = 1000;
int main()
{
   bool prime[MAX];
   prime[0] = false;
   prime[1] = false;
   int i = 2;
   do prime[i++] = true;
   while(i < MAX);
   int j = 2;
   do
      prime[2*j] = false;
   while(++j < MAX/2);
   int p = 3;
   while(p < MAX/3)
   {
      j = 2;
      do prime[p*j++] = false;
      while(p*j < MAX);
      while(prime[p++]==false);
   }
   for(int i = 0; i < MAX; ++i)
      if(prime[i])
         cout << i << " ";
}
```

Note that there is a logical error here which can be fixed by changing

```
while(prime[p++]==false);
```

to

```
while(prime[++p]==false);
```



9. The sum of the first *n* cubes is given by $1 + 2^3 + \cdots + n^3 = \left(\frac{n(n+1)}{2}\right)^2$.
Write a complete program that checks this formula by inputting *n* and then computing and comparing the values of both sides of the equation. Do not use the `cmath` library.

```
#include <iostream>
using namespace std;

int main()
{
     int sum = 0, N;
     cout << "\nEnter a positive integer: ";
     cin >> N;
     for(int n=1;n <= N;++n)
          sum += n*n*n;
     cout << "\nsum = " << sum << endl;
     cout << "(" << N << "*" <<N+1 << "/2)^2="
           << (N*(N+1)/2)*(N*(N+1)/2) << endl;
}
Output:

Enter a positive integer: 4
sum = 100
(4*5/2)^2=100
```

10. Write a program that adds the computes the smallest value of *n* so that $1 + \frac{1}{8} + \frac{1}{27} + \cdots + \frac{1}{n^3} > 1.2$
Do not use the `cmath` library.

```
#include <iostream>
using namespace std;

int main()
{
   float sum = 0, n=1;
   while(sum < 1.2)
   {
        sum += 1/(n*n*n++);
   }
   cout << "\nThe smallest n is " << n << endl;
}

Output:
The smallest n is 17
```