CSci 1113
Midterm 1

Name: ________________________________________________

Student ID: ________________________________

Instructions: Please pick and answer any 6 of the 7 problems for a total of 60 points. If you answer more than 7 problems, only the first 6 will be graded. The time limit is 50 minutes. Please write your answers in the space provided. The exam is open book and notes. You may use electronic devices to ONLY look at either an e-book version or electronic notes. You may not use the internet, compiler or any other outside resources. (If you are typing on your keyboard/input device for anything other than ctrl-F to find words in the e-book or notes, this is probably not acceptable.)

Problem (1) [10 points] Chris likes to fail students, so he made a program that makes big ”F”s. Assume there is some variable size to control how many vertical ”F”’s there are (you may assume size is odd). Use size to create the ”F” as shown below (the amount of horizontal ”F”’s are (1+size)/2):

Sample output (size=5):

F F F
F
F F F
F
F

Grade breakdown:
3 points: loop to print F’s horizontally
2 points: horizontal F’s in proper place (top and halfway)
1 point: correct number of horizontal Fs
2 points: vertical F’s correct
2 points: correct number of vertical F’s

Solution:
    int size = 7;
    for(int i=0; i < size; i++)
    {
        if(i == 0 || i == size/2)
        {
            for(int i=0; i < (size+1)/2; i++)
            {

cout << "F ";
}
cout << endl;
}
else
{
    cout << "F" << endl;
}
}
Problem (2) [10 points] Assume a user enters the following (underlined): $12,345,678.9 = my salary
Write C++ statements that will read this input and then store them into five variables called a, b, c, d and e as follows:

Variable "a" stores: $
Variable "b" stores: 12
Variable "c" stores: ,
Variable "d" stores: 345,678.9
Variable "e" stores: = my salary

Grade breakdown:
1.5 points: correct type per variable (a,b,c,d,e)
1.5 points: correct cin
1 point : using getline for e

Solution:
char a, c;
int b; // or double
string d, e;

cin >> a >> b >> c >> d;
getline(cin, e);

Problem (3) [10 points] Write C++ code that allows the user to input a year. A year that is not a century is a leap year if it is divisible by 4. In the Gregorian calendar, centuries (years divisible by 100) are not leap year unless they are divisible by 400 (sound familiar?). In the Revised Julian Calendar, centuries are not leap years, unless they have remainder of 200 or 600 when divided by 900. Display whether the year is a leap year or not based on the Revised Julian Calendar.

Grade breakdown:
1 point: cin an int (double no good with %)
2 points: non-century 4 are leap year
2 points: non-4 years are not leap years
2 points: special leap year case (200’s 600’s)
2 points: non-special centuries not leap year
1 point: only 1 cout happens

Solution: (may ways)
   int x;
   cin >> x;

   if(x%4 == 0){
      if(x%100 == 0){
         if(x%900 == 200 || x%900 == 600){
            cout << "Leap year" << endl;
         }
      } else{
         cout << "Not a leap year" << endl;
      }
   } else{ // x%4==0 && x%100 != 0
      cout << "Leap year" << endl;
   }
   else { // x%4 != 0
      cout << "Not a leap year" << endl;
   }
Problem (4) [10 points] Allie alternates days eating pizza and working out. If she eats pizza, she eats 3 slices and each slice is 30g of carbs. When she works out, she burns 20 percent of her carbs. Assume there is a variable time that is initialized. Find out how many carbs Allie has after time days of doing this routine. Your program only needs to output her final carb value. Example calculation (for input of time=5, you should show: 219.6):

Day 1 = eat 3 slice (90g),
Day 2 = workout (72g),
Day 3 = eat 3 slices (162g),
Day 4 = workout (129.6),
Day 5 = eat 3 slices (219.6g),
...

Grade breakdown:
2 points: variable to store carb amount (double) and initialize
2 points: loop over time
2 points: every other add 90
2 points: every other remove 20%
1 point: final cout
1 point: correct answer

Solution:
```cpp
double hungryAllie = 0;
for(int i=0; i < time; i++) {
    if(i%2 == 0) {
        hungryAllie += 90;
    }
    else {
        hungryAllie *= 0.8;
    }
}
cout << hungryAllie << endl;
```
**Problem (5) [10 points]** Write C++ code that reads a number in, then displays whether or not it is a Fibonacci number. The definition of a Fibonacci number is that it is the sum of the previous two Fibonacci numbers ($F_n = F_{n-1} + F_{n-2}$). Here is code to generate $n$ Fibonacci numbers:

```cpp
int f0=0;
int f1=1;
for(int i=0; i < n; i++)
{
    int next = f1+f0;
    f0=f1;
    f1=next;
    cout << next << " ";
}
```

Grade breakdown:
1 point: initialize and read in an int
1 points: initialize fibonacci numbers
4 points: non-infinite loop that will reach up to the desired number
3 point: correct answer
1 point: if-statement

**Solution:**

```cpp
int desired;
 cin >> desired;

int f0=0;
int f1=1;
while(f1 < desired) {
    int next = f1+f0;
    f0=f1;
    f1=next;
    cout << next << " ";
}
if(f1 == desired) {
    cout << "Is Fibonacci" << endl;
}
else {
    cout << "Nay" << endl;
}
```
Problem (6) [10 points] Dan is feeling super lazy and wants to make a imperial to metric unit converter. There are 12 inches in 1 foot. One foot is 0.3 meters. Assume the user will input feet with a ‘ unit marker and inches with a ” (as shown in the sample output). Make a program that converts to meters.

Sample output (user input is underlined):
6’11"
2.075

Grade breakdown:
3 points: reading feet and inch part correct
3 points: reading and ignoring the ‘
4 points: displaying correct answer

Solution:
```cpp
int f,i;
char d;
cin >> f>>d>>i;

cout << (f+i/12.0)*0.3 << endl;
```

Problem (7) [10 points] Find 3 possible places for errors in the following code. The lack of #include and using namespace is not an error. You can assume it is in main(). Explain specifically what causes the error and whether it is a syntax, runtime or logic error.

```cpp
cout << "Here is "TOM" 33 times:" << endl;
int nums = 0;
while(nums < 100) {
    if(nums%3 == 0) {
        cout << "T";
```
}  
else if(nums%3 != 2) {  
cout << '0';  
}  
else {  
cout << "M\n";  
}  
nums++;  
}  

Grade breakdown:  
5 free points:  
5 points: if they find the 1 error, full points (-2 points for incorrect type)  
(oops, accidentally didn’t "break" the code)  

Solution:  
logic, loop prints out extra "T"