Lecture 3
Fall 2018
Prof. Zhang

Last week:

1. Three aspects of programming:
   a) programming language (syntax and semantics)
   b) problem solving skills (divide and conquer, step-wise refinement, …)
   c) software engineering (coding style — indentation, comments; tools — debugger, profiler; makefiles; …)

2. Quick overview of C++ language syntax and semantics:
   a) C++ program structure: include directive, main function, …
   b) C++ statement
   c) C++ expression, operator precedence rules and association rules

3. (Hand) Tracing a C++ program
   a) finger/arrow pointing to current statement
   b) labelled box to represent variables
   c) more later: passing parameters during function calls

4. During lab:
   a) log on to storm (from lab computer, own laptop…)
   b) submitting labs
   c) vi/emacs for editing, g++ for compiling, and execution/testing
   d) importance of testing

Today:

1. lab1 review

2. lab2 preview

3. Array and functions review (slides) : pass-by-value vs pass-by-reference; swap function

1. Lab1 review

```
#include <iostream>
#include <assert.h> //include this in order to use assert()…
using namespace std;


//precondition: num is between 0 and 9
//postcondition: the english name of num is displayed in
terminal
void PrintOnes (int num);
```

```
//precondition: num is between 11 and 19
//postcondition: the english name of num is displayed in
terminal
void PrintTeen (int num);

//precondition: num is 10, 20, ..., and 90
//postcondition: the english name of num is displayed in
terminal
void PrintTens (int num);


// precondition: num has a value between 0 and 999
// postcondition: the English name of the num is displayed in
terminal (standard output)
void PrintNumber (int num);


int main()
{
        int number = 0;
        cout << "Enter a number ";
        cin >> number;
        PrintNumber(number);

        return 0;

}

// precondition: num has a value between 0 and 999
// postcondition: the English name of the num is displayed
void PrintNumber (int num)
{
        cout <<"PrintNumber " << num << "\n \n";

        assert (num<=999); //checking precondition
        //if the condition num<=999 is false, the whole
    //program aborts and display that this assertion fails


        // divide the numbers by relevant digits to get the
relevant answers
        int hundred = num /100 ;
        int tens = //(num % 100);   //Misleading Names
                   num / 10 % 10; //set tens to the digit in
                                //10-th place
        int ones = (num % 10 );
//A check-point: making sure digits are properly extracted…
        cout <<“checking: hundred=“<< hundred
        <<“; tens=“<<tens<<“; ones=“<<ones<<endl;



//Draw a flowchart about what we want to do first…
  //Please see the last page for the flowchart
  // Rewrite the rest of this function based upon the flowchart

~~//only display hundred if it is completly divisible by 100 --
misleading comment ~~
~~        if (hundred > 0)~~
~~        {~~
~~                PrintOnes(hundred);~~
~~                cout << " Hundred “; ~~
~~       }~~


~~        // printing out the teens in the similar manner~~
~~        if ((tens > 10)&& (tens < 20))~~
~~        {~~
~~                PrintTeen(tens);~~
~~                //setting tens to zero because there is no need
for it anymore~~
~~                tens = tens - tens ;~~

~~        }~~

~~        //if it not teens than start displaying tens by minusing
the ones~~
~~        tens = tens - ones ;~~

~~        if ((tens == 10)|| (tens > 19));~~
~~                PrintTens (tens);~~

~~        if (ones > 0);~~
~~        {~~
~~                        PrintOnes (ones);~~
~~        }~~

~~        cout <<endl;~~
}
```

```
void PrintOnes (int num)
//precondition: num is between 0 and 9
//postcondition: the english name of num is displayed in
terminal
{
        string Names[10]={"Zero", "One", "Two", "Three", "Four",
"Five", "Six", "Seven", "Eight", “Nine"};
        cout << Names[num];

}



void PrintTeen (int num)
//precondition: num is between 11 and 19
//postcondition: the english name of num is displayed
{
    //Better to use nested if statement here; or use an array
    // as lookup table
    string teensNames[9]={“Eleven”, “Twelve”, “Thirteen”,
“Fourteen”, “Fifteen”, “Sixteen”, “Seventeen”, “Eighteen”,
“Nineteen”};

     assert (num>=11 && num<=19);
     cout << teensNames[num-11];
/*
        if (num == 11)
                cout << "Eleven ";
        if (num == 12)
                cout << "Twelve ";
        if (num == 13)
                cout << "Thirteen ";
        if (num == 14)
                cout << "Fourteen ";
        if (num == 15)
                cout << "Fifteen ";
        if (num ==16)
                cout << "Sixteen ";
        if (num ==17)
                cout << "Seventeen ";
        if(num == 18)
                cout << "Eighteen ";
        if (num ==19)
                cout << "Nineteen “;
*/
}
```

```
void PrintTens (int num)
//precondition: num is 10, 20, ..., and 90
//postcondition: the english name of num is displayed in
terminal
{
        if (num == 10)
                cout << "Ten ";
        if (num == 20)
                cout << "Twenty ";
        if (num == 30)
                cout << "Thirty ";
        if (num == 40)
                cout << "Fourty ";
        if ( num == 50)
                cout << "Fifty ";
        if ( num == 60)
                cout << "Sixty ";
        if (num == 70)
                cout << "Seventy ";
        if (num == 80)
                cout << "Eighty ";
        if (num == 90)
                cout << "Ninety ";

}
```

2. Lab2 Preview: a few pointers

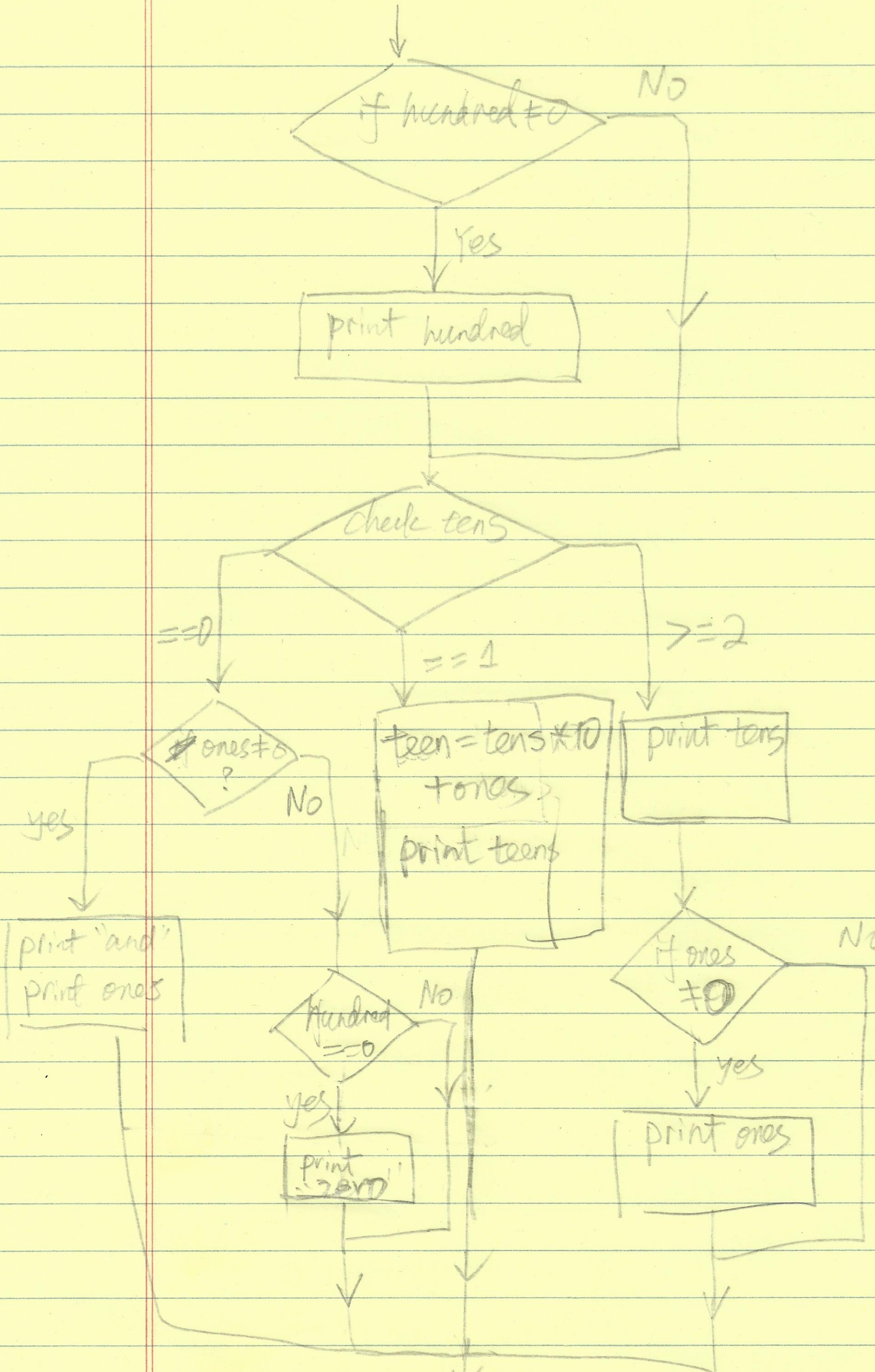
a) Top-down approach to implement the decision tree.
b) Common logic error when testing some condition as follows:

two strings are same if **all characters** in the two string matche.

i.e.,
two strings are not the same if **some characters** do not match.

A number n is prime if it cannot be divided by all numbers from 2 to n-1.

i.e., a number is not prime if it can be divided by one number from 2 to n-1.


if hundred ≠ 0
No
Yes
print hundred
check tens
==0
==1
>=2
~~if~~ ones ≠ 0 ?
yes
No
print "and"
print ones
teen = tens * 10 + ones
print teens
print tens
hundred == 0
No
yes
print "zero"
if ones ≠ 0
No
yes
print ones