CS1A - Exam Review

- Intro to Programming
- Style Requirements
- Basic Input & Output
- Selection
- Repetition
- Testing
- Advanced Selection

FORMAT

- Bring a scantron
- Some T/F
- Some Mult Choice
- Some Problem Solving
- NOT open notes/book

Avoiding Test Anxiety

Get a good nights rest
- I know this is tough, but you don't think as well without sleep

Don't skip a meal before an exam
- Your brain needs protein → try not to eat a high carb meal

Don't Cram! Pace your studying
- Try not to put it off until the last minute
- If you pace yourself → you will be prepared

Study with classmates so you can compare notes
- don't discuss the exam just before coming in
- their anxiety may impact you

Take deep breaths → relax yourself
- Think positive thoughts → remind yourself that you are prepared

Don't get bogged down on a question
- answer the questions you know quickly → go back to the others

Ask Questions
- Calm yourself before you come in…

Avoid being late

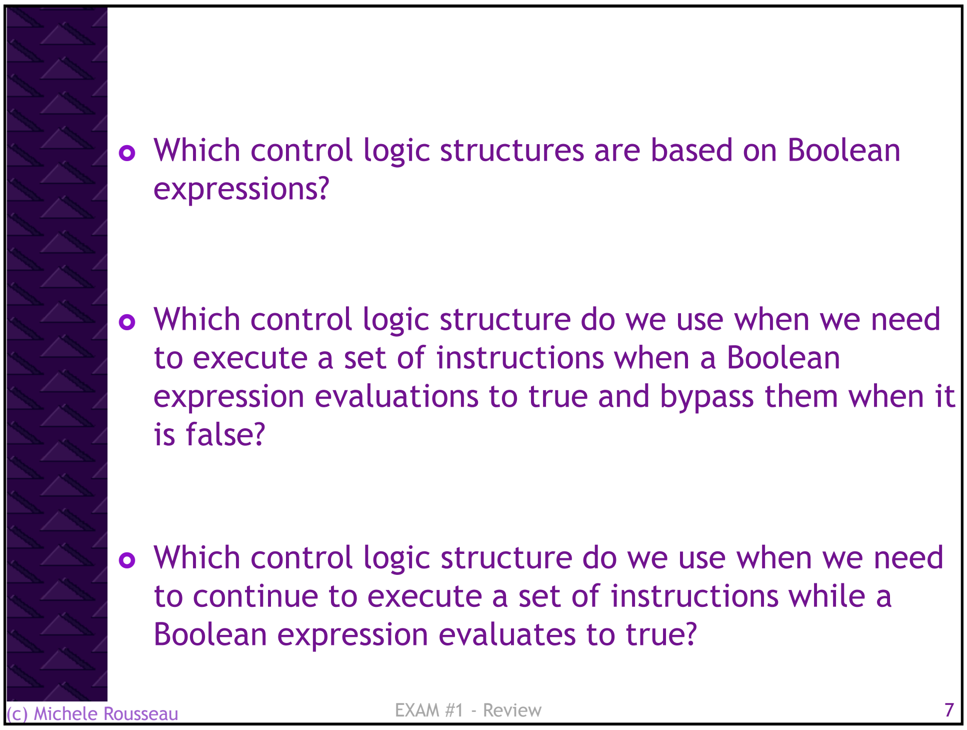
Programming Basics

CS1A

You should know…
- how to declare identifiers
- when to use different data types
- Charts and algorithm development
  - HIPO
  - Flowchart
  - Pseudocode
  - Deskchecks

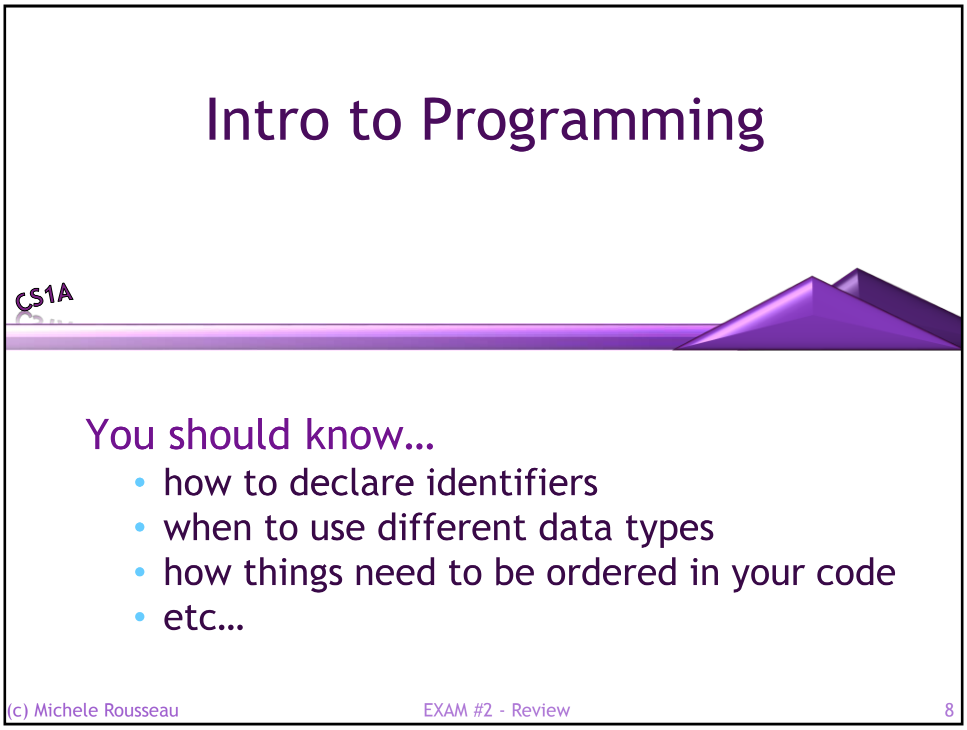
- What diagram have we used that represents the top down design of the basic program modules and how they are related?
- What diagram have we used that depicts the flow of an algorithm using specific symbols to indicate various programming techniques?
- What do the three modules in the 2nd layer of a structure chart represent?
- When do we stop refining in a HIPO chart?

- When do we have to initialize a variable.
- How do we define a value for a variable?
- How do we use a value stored in a variable
- What are the 3 control logic structures?



- Which control logic structures are based on Boolean expressions?
- Which control logic structure do we use when we need to execute a set of instructions when a Boolean expression evaluations to true and bypass them when it is false?
- Which control logic structure do we use when we need to continue to execute a set of instructions while a Boolean expression evaluates to true?

(c) Michele Rousseau EXAM #1 - Review 7



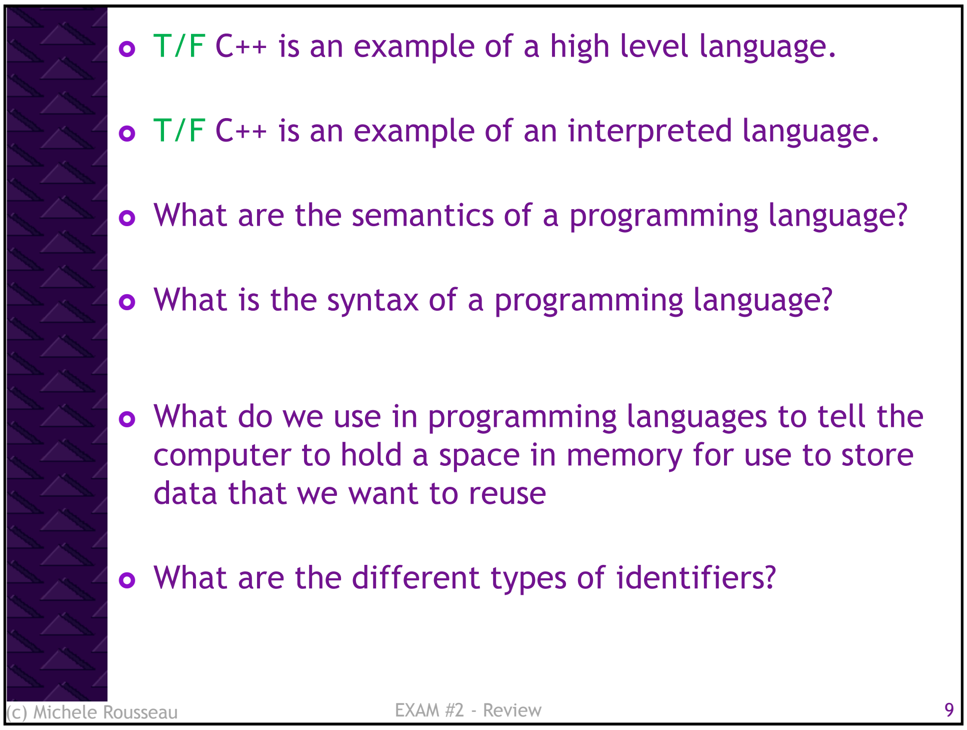
Intro to Programming

CS1A

You should know…

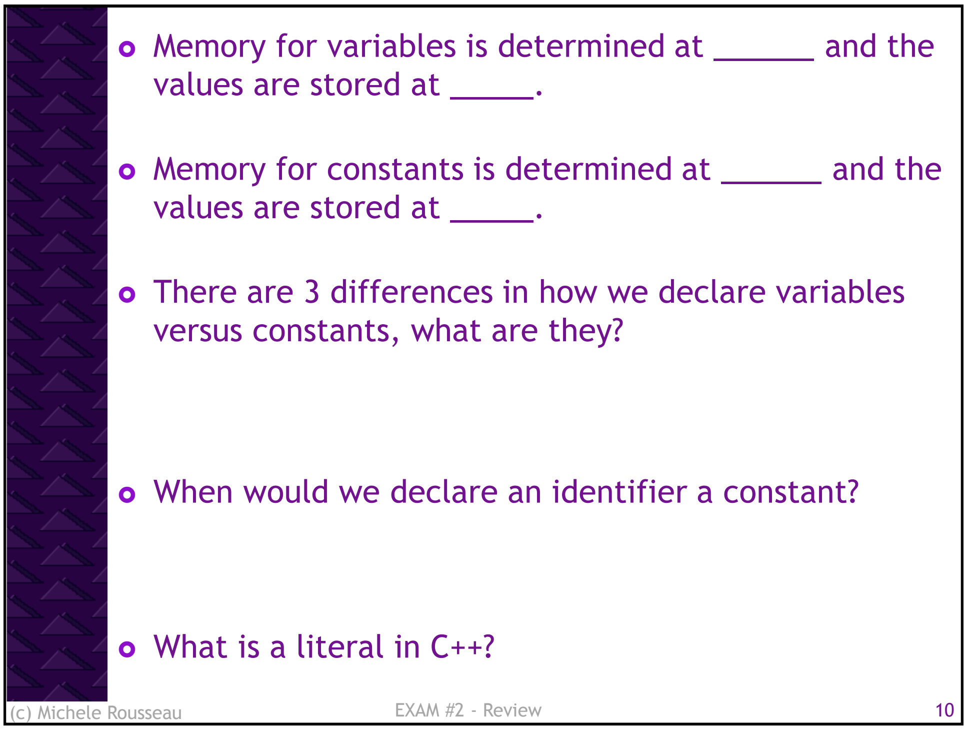
- how to declare identifiers
- when to use different data types
- how things need to be ordered in your code
- etc…

(c) Michele Rousseau EXAM #2 - Review 8



- T/F C++ is an example of a high level language.
- T/F C++ is an example of an interpreted language.
- What are the semantics of a programming language?
- What is the syntax of a programming language?
- What do we use in programming languages to tell the computer to hold a space in memory for use to store data that we want to reuse
- What are the different types of identifiers?

(c) Michele Rousseau EXAM #2 - Review 9

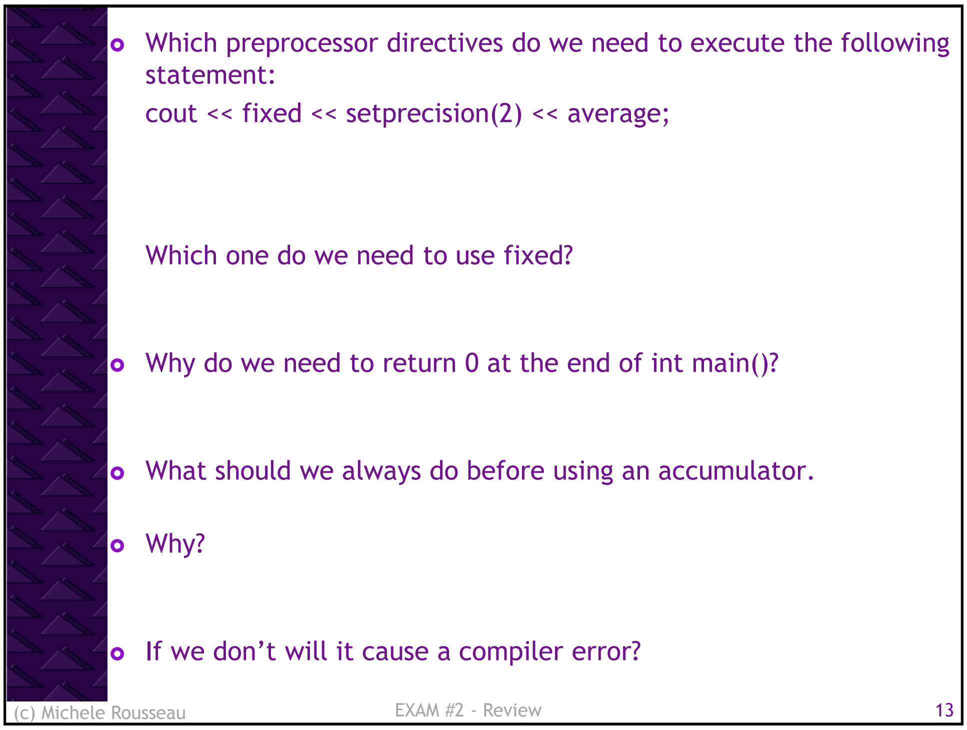


- Memory for variables is determined at _______ and the values are stored at ______.
- Memory for constants is determined at _______ and the values are stored at ______.
- There are 3 differences in how we declare variables versus constants, what are they?
- When would we declare an identifier a constant?
- What is a literal in C++?

(c) Michele Rousseau EXAM #2 - Review 10

- What are the 2 things the compiler needs to know when we declare an identifier
    1.
    2.
- How do we provide this information to the compiler
- What is the difference between ‘A’ & “A” in C++
- What is the difference between ‘A’ and A in C++
- How would we declare a c-string variable which could store a name of up to and including 10 characters?
- How would we declare a constant named A which would store the single value ‘A’?

- What data type would we use to store the average of a sum of integers?
- What data type would we use to store a counter?
- What data type would we use to store an accumulator?
- What data type would we use to store a letter grade?
- Where in a C++ program do we put the declaration section?
- Where in a C++ program do we put the preprocessor directives



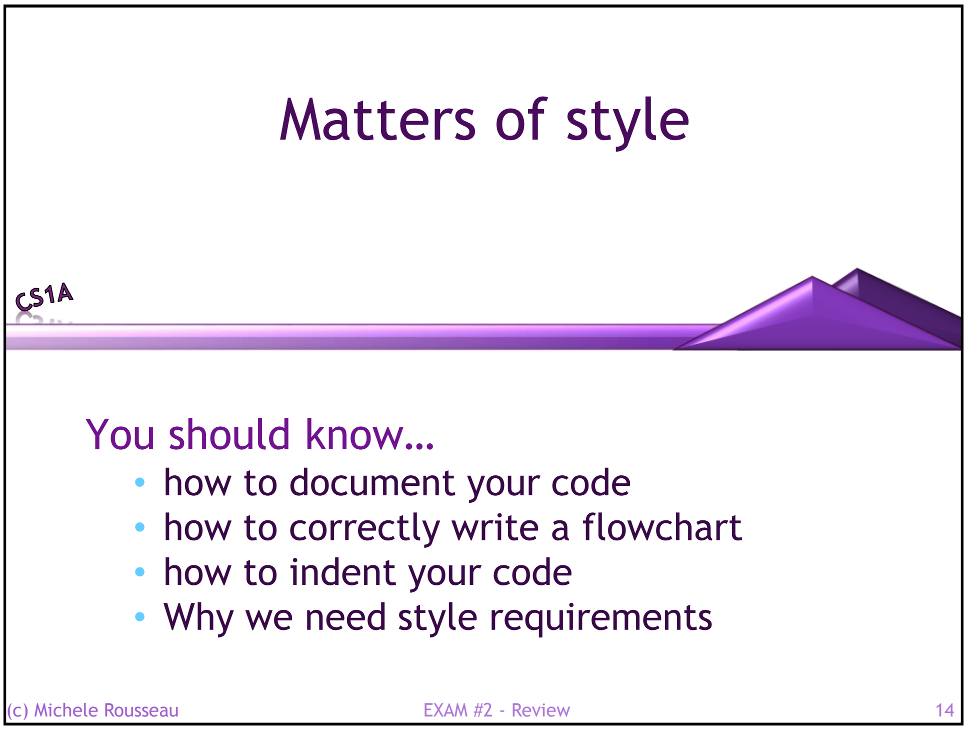
- Which preprocessor directives do we need to execute the following statement:

  cout << fixed << setprecision(2) << average;

  Which one do we need to use fixed?

- Why do we need to return 0 at the end of int main()?
- What should we always do before using an accumulator.
- Why?
- If we don't will it cause a compiler error?

(c) Michele Rousseau EXAM #2 - Review 13



Matters of style

CS1A

You should know…

- how to document your code
- how to correctly write a flowchart
- how to indent your code
- Why we need style requirements

(c) Michele Rousseau EXAM #2 - Review 14

- Why do we have style requirements?

- T/F The following is an example of a correctly written data table:

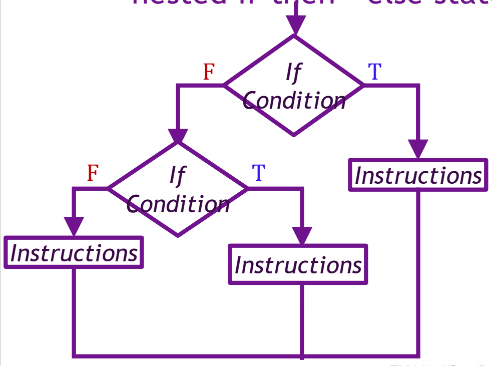
```
int firstNum;  // first value to average - INPUT
int secondNum;  // second value to average - INPUT
float average;  // average of three values – CALC & OUTPUT
```

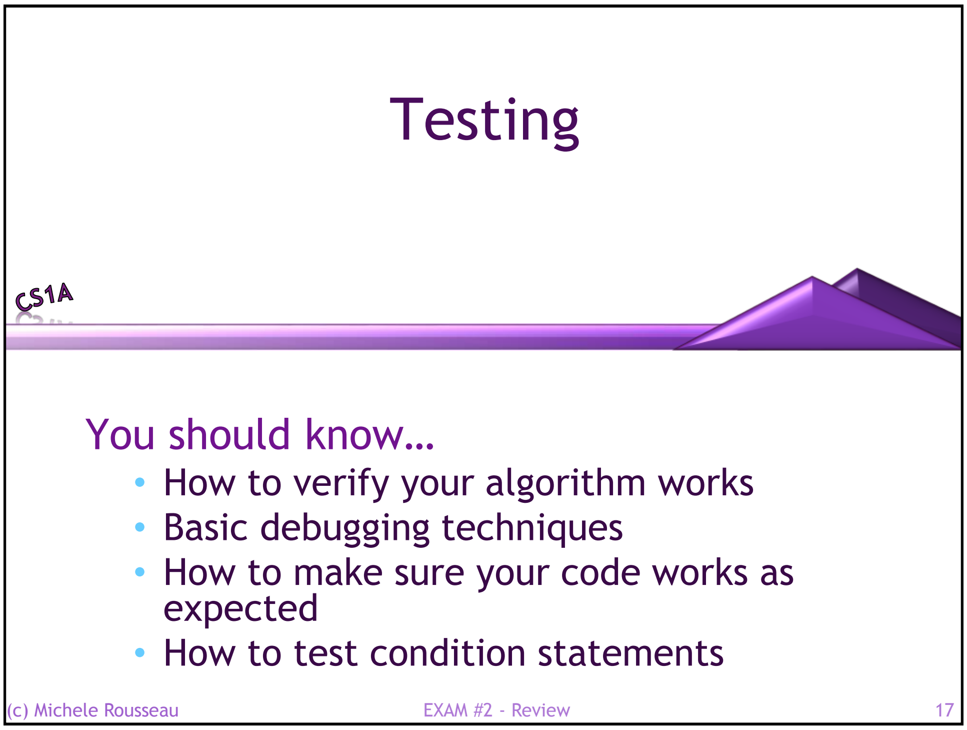
- If you need to describe a section of executable code where should you put the comments?

- T/F It is okay to initialize variables in the declaration section.

- What is the documentation next to the declaration section called?

- What should it tell the reader?
  1.
  2.
- T/F The following represents the proper way to diagram nested if-then – else statements?





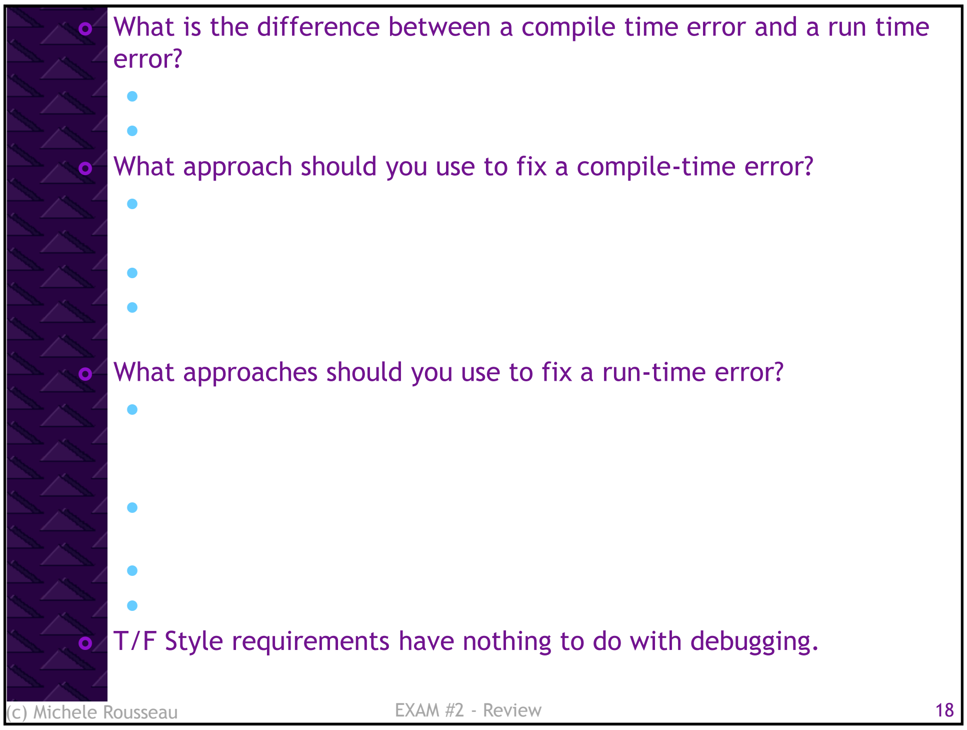
Testing

CS1A

You should know…

- How to verify your algorithm works
- Basic debugging techniques
- How to make sure your code works as expected
- How to test condition statements

(c) Michele Rousseau EXAM #2 - Review 17



- What is the difference between a compile time error and a run time error?
  -
  -
- What approach should you use to fix a compile-time error?
  -
  -
  -
- What approaches should you use to fix a run-time error?
  -
  -
  -
  -
- T/F Style requirements have nothing to do with debugging.

(c) Michele Rousseau EXAM #2 - Review 18

- T/F Style requirements have nothing to do with debugging.
- T/F You should only test your code with the values provided.
- What is a test plan?
- Your code contains the following if statement:

  ```
  if (hours > 40)
  ```

  What values should you test for hours?

Basic Input / Output

CS1A

You should know basic I/O statements plus…

- Output manipulators
  - How to format output in columns
  - How to use the floating point manipulators
- Escape sequences
- Given a segment of code including cout statements be able to determine what the output will look like to a user.
- Given output be able to write the code to produce it
- How the input operators function
  - When do you need ignore?
- Given a problem – be able to write the input section

- What is this operator called <<, and what statement do we use it with?
- What is this operator called >>, and what statement do we use it with?
- In the statement:

  ```
  cout << "Hello you entered!" << someInt;
  ```

  What is "Hello you entered!"

  What is someInt?
- How would you output a single quote?

- How would you specify how a column of output as 5 characters?
- How would the following statement execute?

  ```
  val = 341.2576
  cout << setprecision (3) << val;

  cout << setprecision (3) << fixed << val;

  val 2 = 32;
  cout << setprecision (3) << showpoint << val2;
  ```
- Be sure you understand how the manipulators work together.

- MAKE SURE YOU UNDERSTAND cin.getline, cin.get, & cin.ignore!
- Will this code segment work correctly?

```
int someInt;
char name [25];

cout << “Enter int: “;
cin   >> someInt;

cout << “Enter name: “;
cin.getline(name, 25);
```

- How can you fix it?

Arithmetic

CS1A

You should know…
- Order of precedence
- Type coercion
- Type casting

- What is mixed mode arithmetic
- T/F Integers and floating point #s are stored differently
- Is this type coercion or type casting?

```
avg = (int1 + int2) / 2.0;
```

- What will be in the memory location average after this is executed?

```
 float average;
 inum1 = 3;
 inum2 = 7.75;
 average = (inum1 + inum2) / 20;
```

- How will this differ from?

```
average = (inum1 + inum2) / 20.0;
```

- How would you typecast average = (inum1 + inum2) / 20; to get the correct answer?

Selection

CS1A

You should know…

- how to flowchart each statement
- How to nest them
- Write the code from pseudo code or flowchart
- When to use them
- How to write conditional statements
- if-then-else-if and switch statements

- What are the 3 basic control logic structures?
- Give an example of a one-way decision statement
- Give an example of a two-way decision statement
- Give an example of a multi-way decision statement
- What is the difference between a primary and secondary decision

- Rewrite the following code-segment as a conditional operator

```
int age;
bool regStat;   ….

if (age >= 18)
{
        if (regStat)
        {
               cout << "You can vote";
        {
        else
        {
            cout << "not registered";
        }
}
else
{
        cout << "too young";
}
```

Which statement should you use?

- Suppose we want to output "you are over a century!" when age is greater than 100.
- Suppose we want to output the cardinal directions based on a char ('W' - "west", 'E' - "East", …)
- Suppose we want to output the sign of an integer (eg. '+' if int1 is > 0, 0 if it is equal to 0 and '-' if it is negative.
- Suppose we want to output "go to the beach" when the weather is good and output "stay home and watch a movie" when it is not

Be able to write any of these statements

Repetition

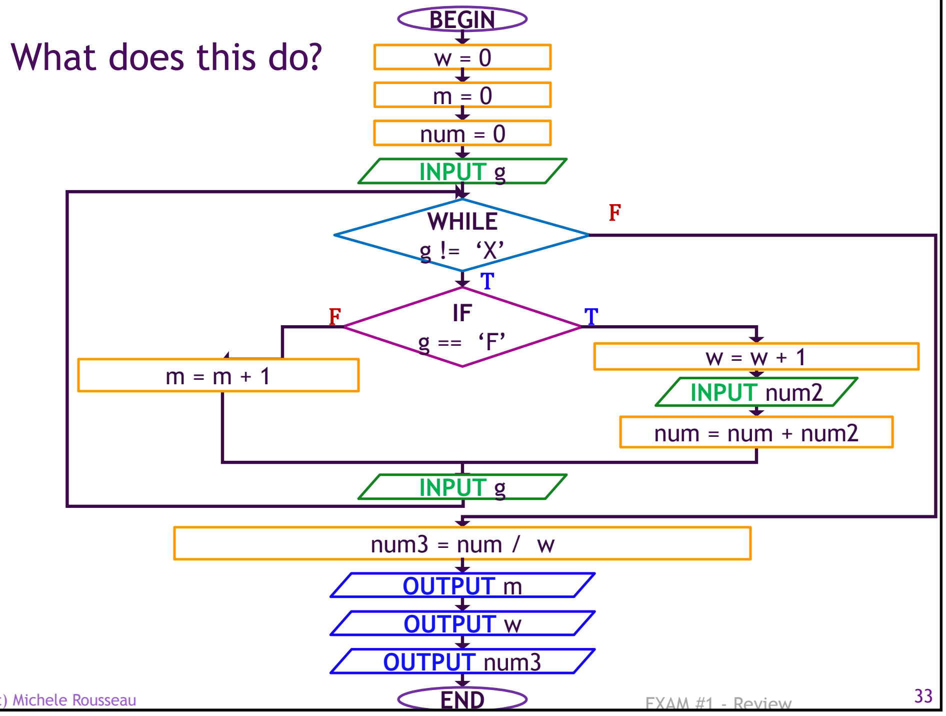
CS1A

You should know…
- how to flowchart each statement
- How to nest them
  - With loops and selection statements
- When to use them

- What are the three loop structures?
- Which Loop(s) are counter based?
- Which Loop(s) are pre-test loops?
- T/F IF-THEN statements can have the instructions on the False side?
- What 3 steps need to be taken with respect to every loop?
  1 - Initialize the LCV
  2 - Check the LCV
  3 - Change the LCV
  - When should these happen for each loop?
- Be Prepared to be able to draw a HIPO chart, flowchart, or write pseudocode or code
- Be prepared to be able to perform a desk check on an algorithm

- Know when to use a While vs. For Loop each.
- Know the basic structure for loops processing user input.
- When do you need to initialize your variables?
- What is an accumulator?
- What is a counter?
- What is an infinite loop

What does this do?



- Given the following code segment

```
a = 0;
y = 0;
z = 0;
cout << "Enter a: ";
cin   >> a;
while (a > -1)
{
      if (a > 10)
       {
           y = y + 1;
       }
      z = z + a;

      cout << "Enter a: ";
      cin >> a;
}
```

What is the LCV?

Where is it initialized, checked and changed?

What is/ are the accumulators?

What is/are the counters?

What does this do?

Developing an algorithm

What steps are involved in developing an algorithm that includes a loop?

Determining the LCV

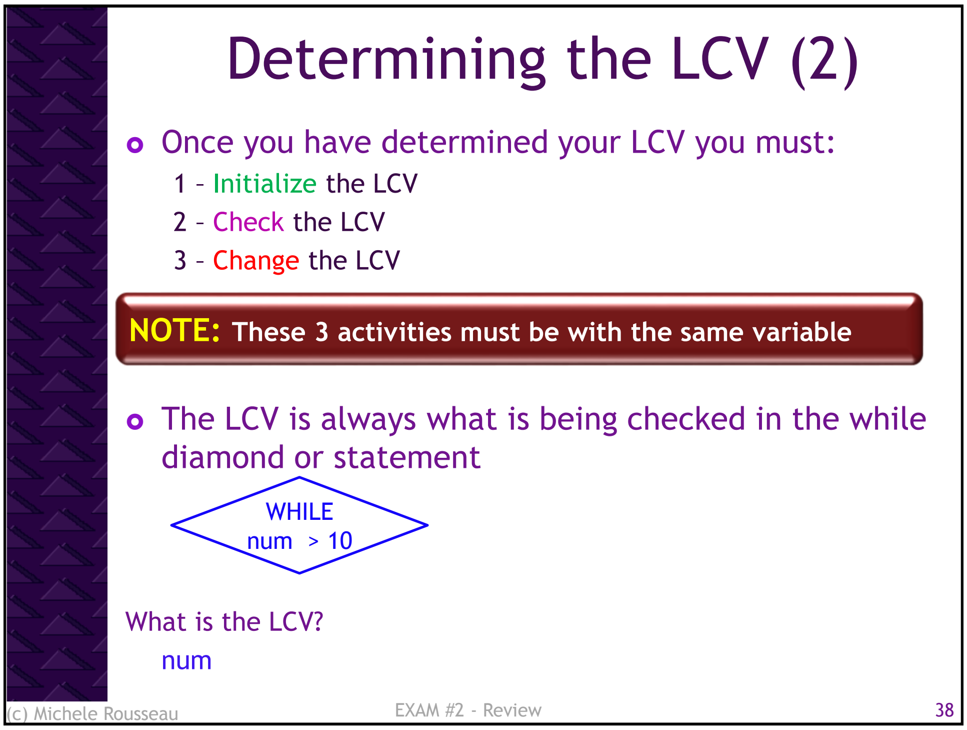
- It is important with while and do while loops to choose the correct LCV
- For while & do while loops you need to determine when you want the loop to execute and when you want it to stop

Example Program Requirement

Allow the user to enter in numbers until they enter 0.

What should use as an LCV?

- Under what condition should the loop execute
- When should it stop?

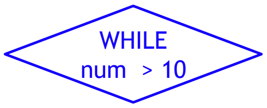


Determining the LCV (2)

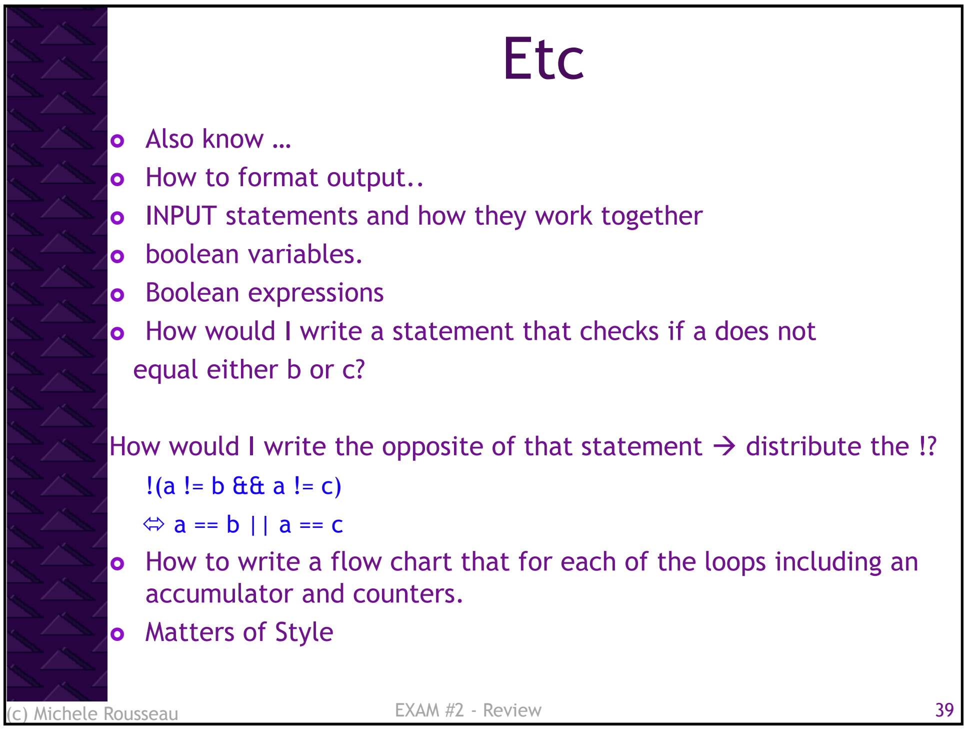
- Once you have determined your LCV you must:
  1 - Initialize the LCV
  2 - Check the LCV
  3 - Change the LCV

NOTE: These 3 activities must be with the same variable

- The LCV is always what is being checked in the while diamond or statement



What is the LCV?
num



Etc

- Also know …
- How to format output..
- INPUT statements and how they work together
- boolean variables.
- Boolean expressions
- How would I write a statement that checks if a does not equal either b or c?

How would I write the opposite of that statement → distribute the !?
!(a != b && a != c)
⇔ a == b || a == c
- How to write a flow chart that for each of the loops including an accumulator and counters.
- Matters of Style