Computer Programming
Lecture 3

Hung-Yu Wei
Department of Electrical Engineering
National Taiwan University

Review Question I

```
X=3;
Y=7;
Z=(X==Y);
```

- What are the values of X, Y, Z?
- You also need to know what are the memory operations when you do these calculations.

Review Question II

```
X=3;
Y=7;
Z1=Y/X;
Z2=Y%X;
```

- What are the values of X, Y, Z1,Z2?

What will we learn today

- How do you control the “flow” of your program?
  - Selection
  - Repetition
- If
- If … else
- while
- More about UML

Introduction: solving problem and writing program

- Before writing a program
  - Have a thorough understanding of problem
  - Carefully plan your approach for solving it
- While writing a program
  - Know what “building blocks” are available
  - Use good programming principles

Concept: algorithms

- Computing problems
  - Solved by executing a series of actions in a specific order
- Algorithm is a procedure determining
  - Actions to be executed
  - Order to be executed
  - Example: cooking/recipe
- Example
  - Sorting algorithm
  - Computing Fibonacci numbers
- Program control
  - Specifies the order in which statements are executed

Concept: pseudocode (fake code)

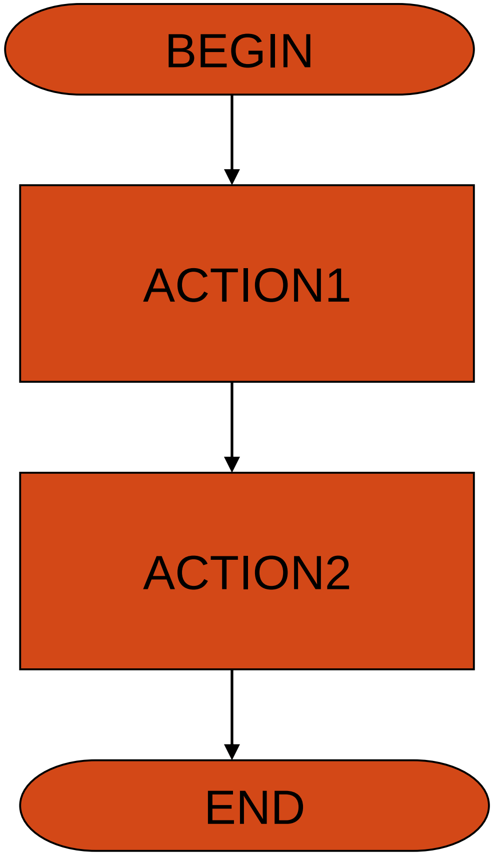
- Pseudocode
  - Artificial, informal language used to develop algorithms
  - Similar to plain English
- Not executed on computers
  - Used to think out program before coding
    - Easy to convert into C++ program
  - Only executable statements
    - No need to declare variables (optional)
- Help you to analyze/design/program

Concept: goto statement

- It is BAD!
  - Jumping between the lines is unstructured
- Structured programming
  - Avoid using goto statement
  - Also known as “goto elimination”
  - C is a structured language

Flowchart

- Graphical representation of an algorithm



Types of statement

- Sequence statement
- Selection statement
  - Single selection statement: If
  - Double selection statement: If … else
  - Multiple selection statement: switch
- Repetition statement
  - Also known as loops or looping statement
  - While
  - Do … while
  - For

Review: if statement

- Pseudocode

```
If student’s grade is greater than or equal to 60
    Print “Passed”
```

- C++

```
if (grade >= 60)
    cout << “Passed”;
```

while

- A repetition statement
- Example: pseudocodes

```
While there are more items on my shopping list
        Buy the next item and update the list
```

- Syntax

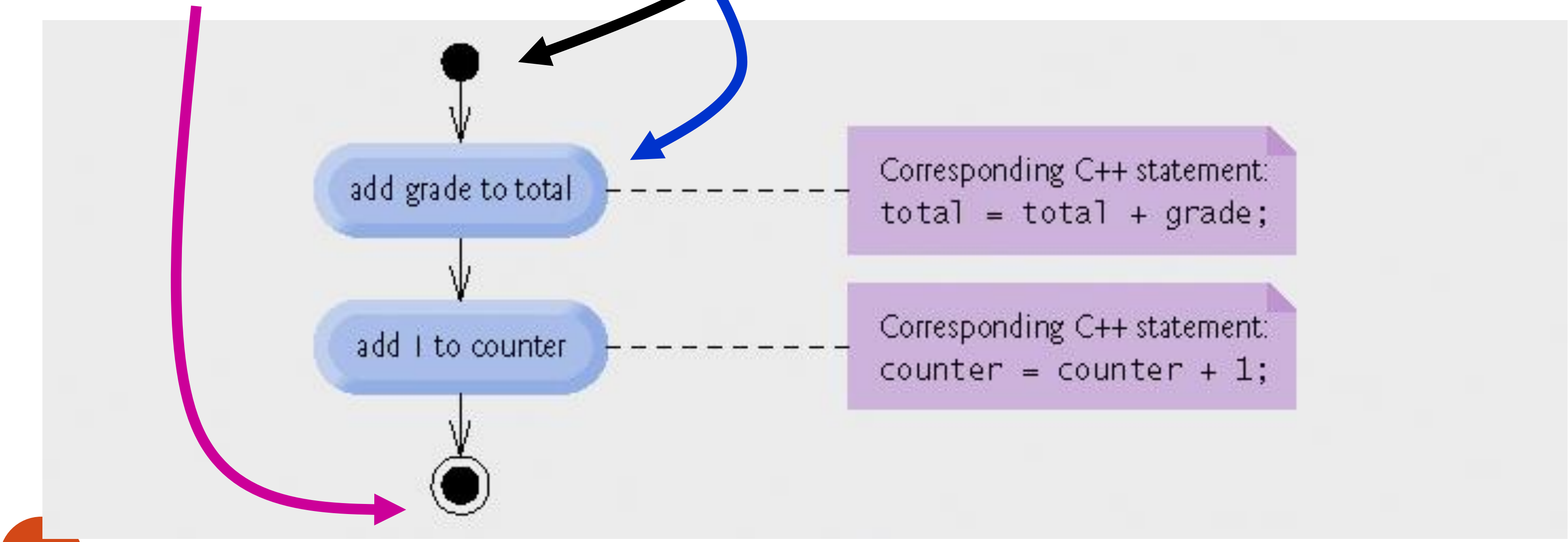
```
while (condition)
        statement to do;
```

UML

- Unified Modeling Language (UML)
- An industry standard for modeling software systems
- Useful for object-oriented programming

Sequence-structure activity diagram

- Initial state
- Action state
- Final state



Corresponding C++ statement:

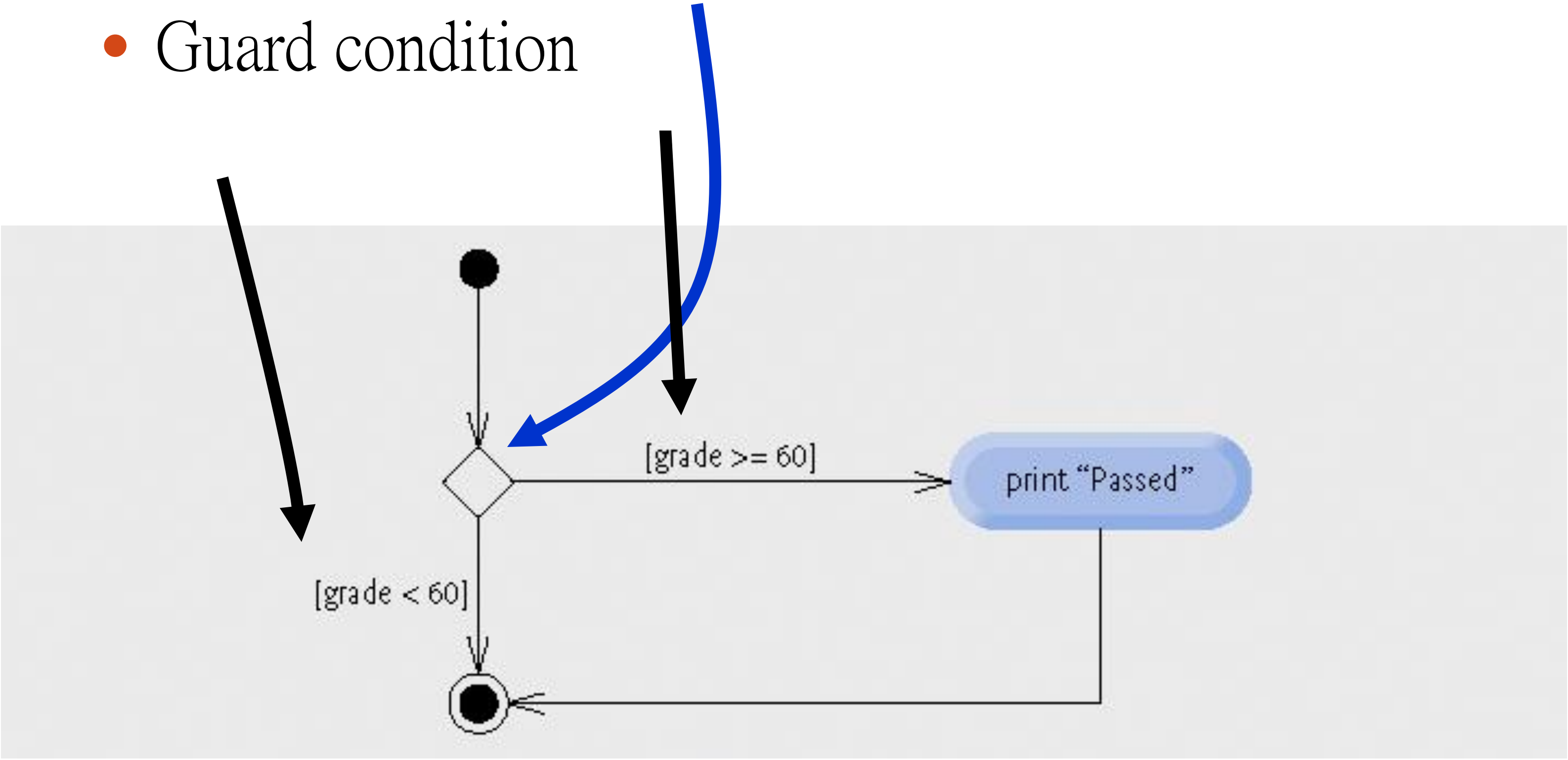
```
total = total + grade;
```

Corresponding C++ statement:

```
counter = counter + 1;
```

If single-selection statement activity diagram

- Decision symbol
- Guard condition



if...else double-selection

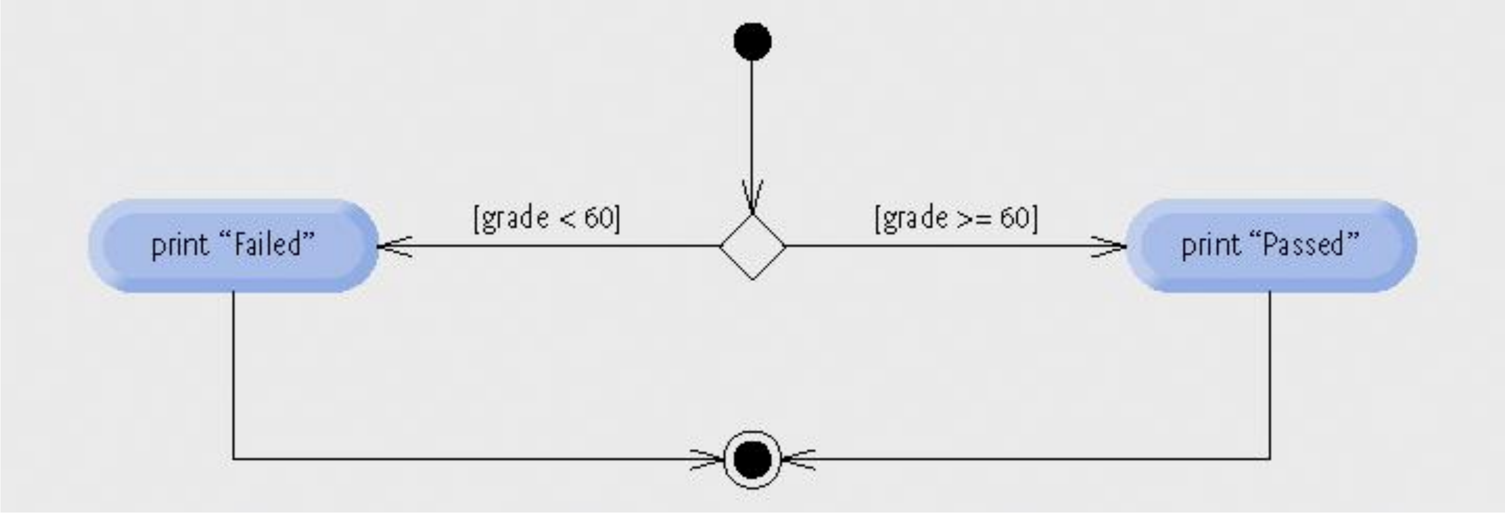
- Pseudocodes

```
If student’s grade is greater than or equal to 60
        Print “passed”
else
        Print “Failed”
```

- C++ codes

```
if (grade >= 60)
        cout << “Passed”;
else
        cout << “Failed”;
```

if…else double-selection statement activity diagram.



?:

- Conditional Operator
- Similar to the if …else syntax
- Example
  - Cout << (grade >=60 ? “Passed” : “Failed”);
  - Grade >=60 ? cout << “Passed” : cout << “Failed”;
- Usage is similar to if … else

?:

- Syntax

(condition) ? (actions, if it’s true) : (actions, if it’s false)

- Can be used within 1 line

If (condition)

    (actions, if it’s true)

Else

    (actions, if it’s false)

Nested if … else statement

- Pseudocodes

```
If student's grade is greater than or equal to 90
    Print "A"
Else
    If student's grade is greater than or equal to 80
            Print "B"
    Else
            If student's grade is greater than or equal to 70
               Print "C"
            Else
               If student's grade is greater than or equal to 60
                   Print "D"
               Else
                   Print "F"
```

C++

```
if (studentGrade >=90)
   cout <<"A";
else
   if (studentGrade >=80)
          cout <<"B";
   else
          if (studentGrade >=70)
                    cout <<"C";
          else
                    if (studentGrade >=60)
                              cout <<"D";
                    else
                              cout <<"F";
```

Continued

- C++

```
if (studentGrade >=90)
    cout << “A”;
else if (studentGrade >=80)
    cout << “B”;
else if (studentGrade >=70)
    cout << “C”;
else if (studentGrade >=60)
    cout << “D”;
else
    cout << “F”;
```

- if
- else if
- else

Some tips

- A *nested if…else* statement can perform much faster than a series of single-selection if statements
- In a *nested if… else* statement, test the conditions that are more likely to be true at the beginning of the nested if…else statement.

Dangling-else problem

- An example

```
If (x>5)
{
   if (y>5)
   {       cout << “x,y are greater than 5”;
    }
} else
{
   cout << “x is <= 5”;

}
```

```
if(x>5)
    if(y>5)
        cout << "x, y > 5 \n";
else
        cout << "x <= 5";
```

```
if(x>5)
    if(y>5)
        cout << "x, y > 5 \n";
    else
        cout << "x <= 5";
```

[Continued]Dangling-else problem

- The reality (it is confusing)
- Use { } to make your program clear

```
if (x>5)
   if (y>5)
          cout << “x,y are greater than 5”;
   else
          cout << “x is <= 5”;
```

{ }

- Use { } for clarity and multiple lines of statement in *if…else*

```
If (x>5)
{
   if (y>5)
          cout << “x,y are greater than 5”;
}
else
   cout << “x is <= 5”;
```

Tips

- Always putting the { } in an if…else statement (or any control statement) helps prevent their accidental omission
  - You could put { } before writing any statements within { }

Double Selection: if else and { }

- One statement

```
If (condition)
    one statement;
else
    another statement;
```

- Multiple statements

```
if (condition)
{
    multiple statements;
}
Else
{
    more statements;

}
```

An example with *while*

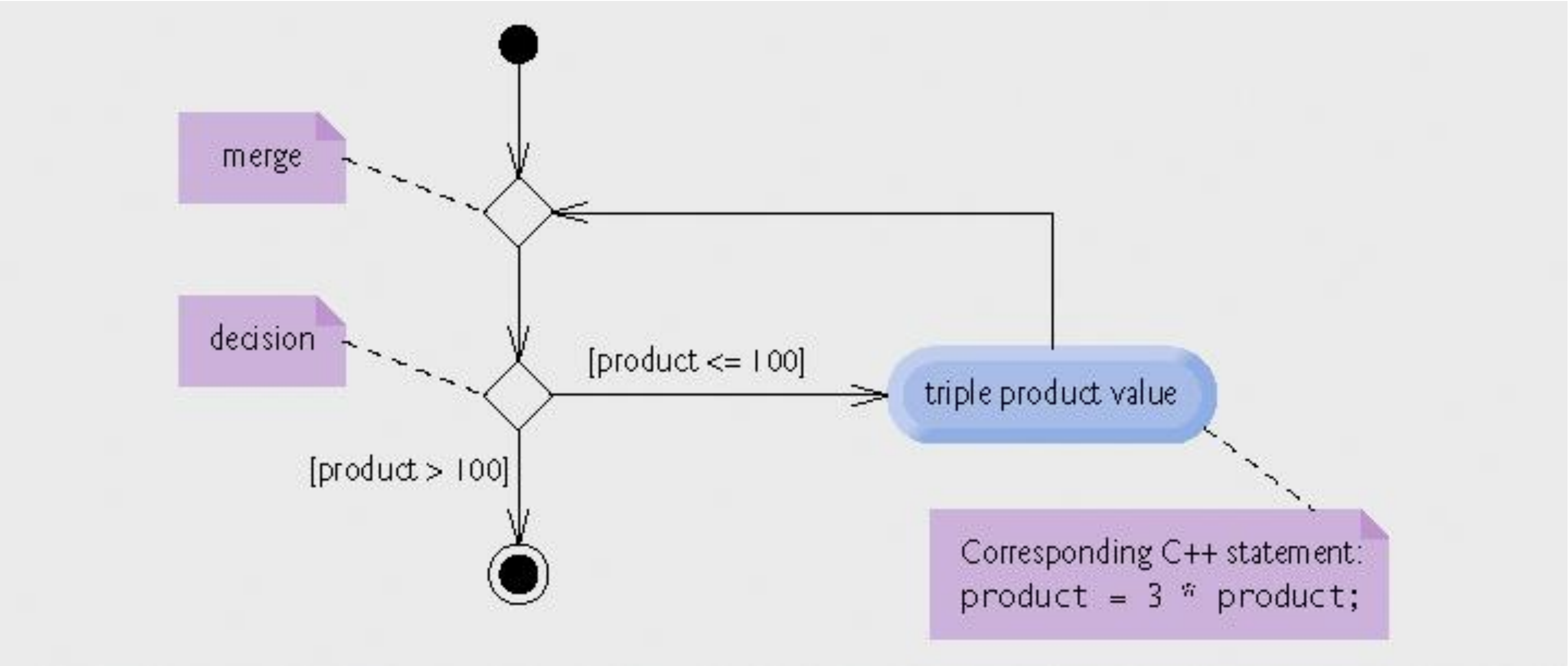
- Example: compute a value which is 3^N and is greater than 100

```
int product =3;

while (product <= 100)
    product=3*product;
```

while repetition statement UML activity diagram

- UML uses the same icon for *merge symbol* and *decision symbol*



Corresponding C++ statement:
`product = 3 * product;`

Be careful with *while*

- infinite loop
  - Repetition statement never terminates
- This can make a program appear to “hang” or “freeze” if the loop body does not contain statements that interact with the user.
- Use *while* loop carefully

What have we learned so far?

- Basic C++ program
- Main program
- Include C++ standard library (e.g. iostream)
- I/O (cin/cout)
- Arithmetic
- Control statement
  - Create your program from pseudocodes to C++
  - If, If…else, While