Computer Programming 143 – Lecture 7
Program Control I

Electrical and Electronic Engineering Department
University of Stellenbosch

Prof Johan du Preez
Mr Callen Fisher
Dr Willem Jordaan
Dr Hannes Pretorius
Mr Willem Smit

Copyright & Disclaimer

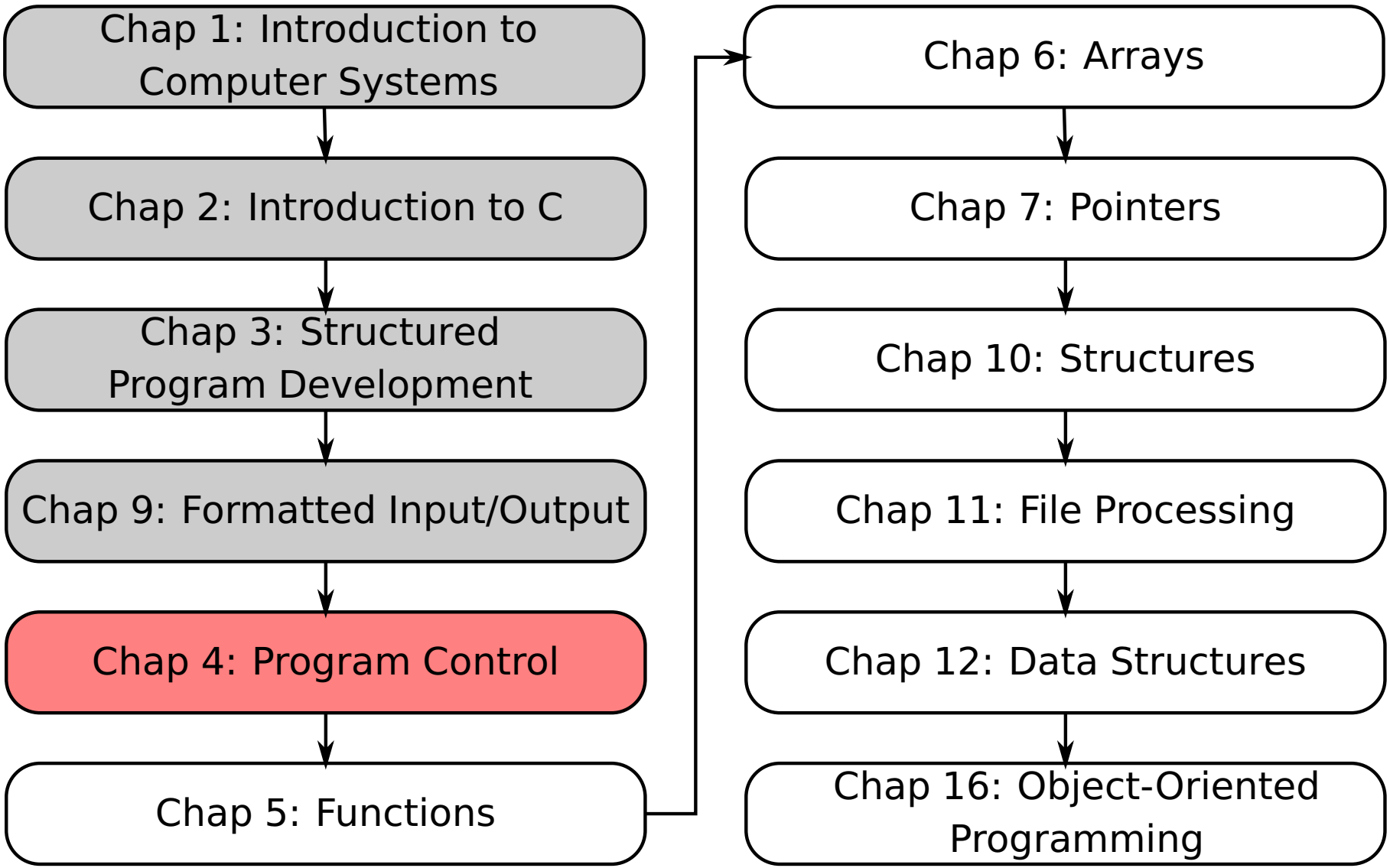
Disclaimer

This content is provided without warranty or representation of any kind. The use of the content is entirely at your own risk and Stellenbosch University (SU) will have no liability directly or indirectly as a result of this content.

The content must not be assumed to provide complete coverage of the particular study material. Content may be removed or changed without notice.

The video is of a recording with very limited post-recording editing. The video is intended for use only by SU students enrolled in the particular module.

Module Overview



Lecture Overview

1. The Essentials of Repetition and Counter-Controlled Repetition (4.1-4.3)
2. The `for` Repetition Statement (4.4-4.5)
3. `for` Examples (4.6)

4.2 The Essentials of Repetition

Loop

- A group of instructions that the computer executes repeatedly, while some condition remains **`true`**

Counter-controlled repetition

- Definite repetition: known beforehand how many times loop will execute
- Control variable used to count repetitions

Sentinel-controlled repetition

- Indefinite repetition: not known beforehand how many times loop will execute
- Sentinel value indicates “end of data”

4.3 Essentials of Counter-Controlled Repetition

Example:

```
int counter = 1;          /* initialization */
while ( counter <= 10 ) /* repetition condition */
{
  printf( "%d\n", counter );
  ++counter;              /* increment : same as counter = counter +1;*/
}
```

Same as:

```
int counter = 0;          /* initialization */
while ( counter < 10 ) /* repetition condition */
{
  ++counter;              /* increment */
  printf( "%d\n", counter );
}
```

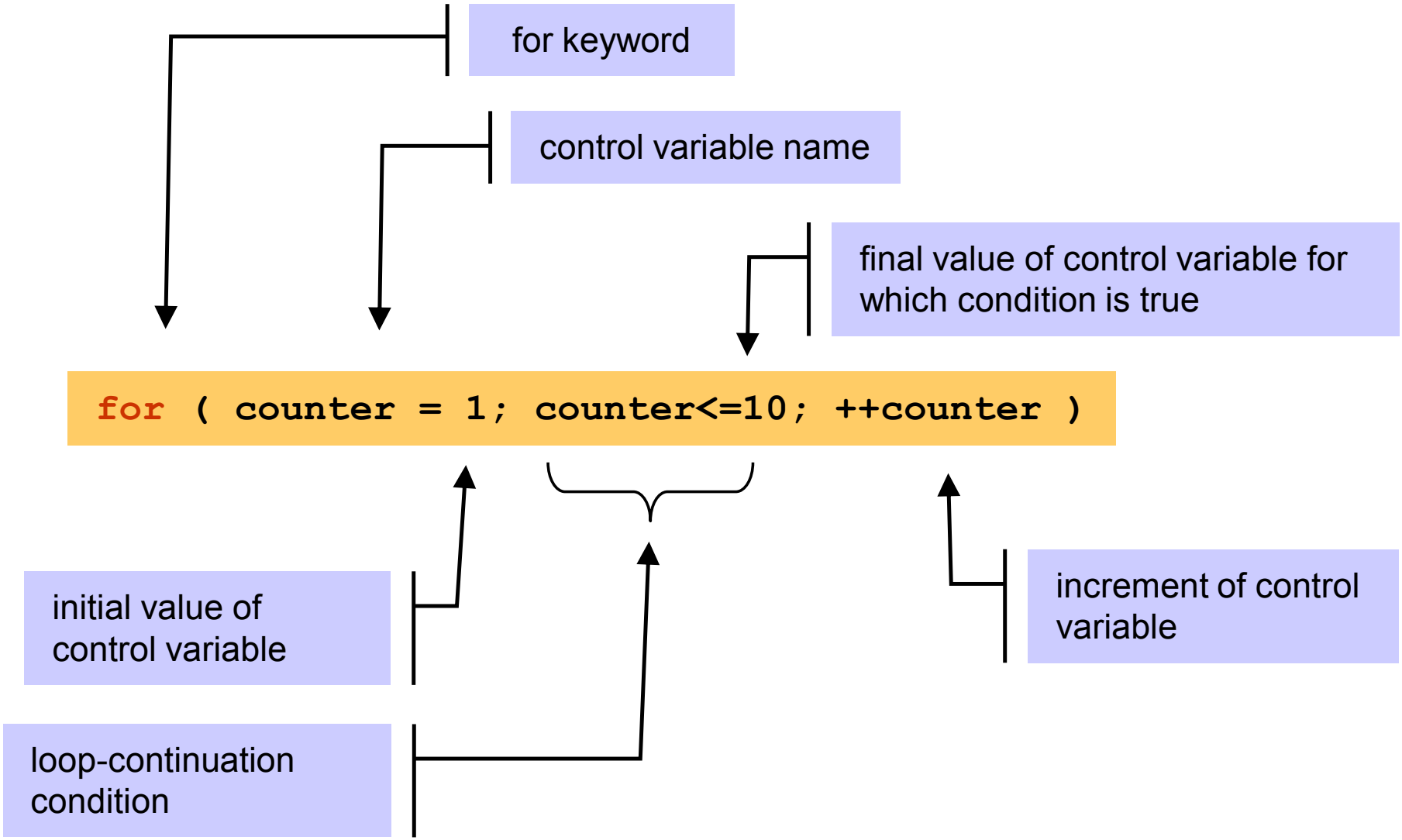
4.3 Essentials of Counter-Controlled Repetition

The statement

```
int counter = 1;
```

- Names the variable **counter**
- Defines it to be an integer
- Reserves space for it in memory
- Sets it to an initial value of 1 (i.e. initialises it to 1)

4.4 The **for** Repetition Statement I



4.4 The **for** Repetition Statement II

Format when using **for** loops

for (*initialization; loopContinuationTest; increment*) {
    statement;
}

Example:

```
int counter;
for( counter = 1; counter <= 10; counter++ ) {
  printf( "%d\n", counter );
}
```

- Prints the integers from one to ten

4.4 The **for** Repetition Statement III

for loops can usually be rewritten as **while** loops

```
initialization;
while ( loopContinuationTest ) {
    statement;
    increment;
}
```

for

```
for(i = 1; i <= 10; i++) {
  printf("%d\n", i);
}
```

while

```
i = 1;
while (i <= 10) {
  printf("%d\n",i);
  i++;
}
```

4.5 The **for** Statement : Notes I

Initialisation and incrementation

- Can be comma-separated lists
- Example:

```
for (i = 0, j = 0;   j+i <= 10;   j++, i++) {
  printf( "%d\n", j+i );
}
```

4.5 The `for` Statement : Notes II

Arithmetic expressions

- Initialisation, loop-continuation, and increment statements can contain arithmetic expressions.

```
x = 2;
y = 10;
for ( j = x; j <= 4*x*y; j += y/x )
```

- is equivalent to

```
for ( j = 2; j <= 80; j += 5 )
```

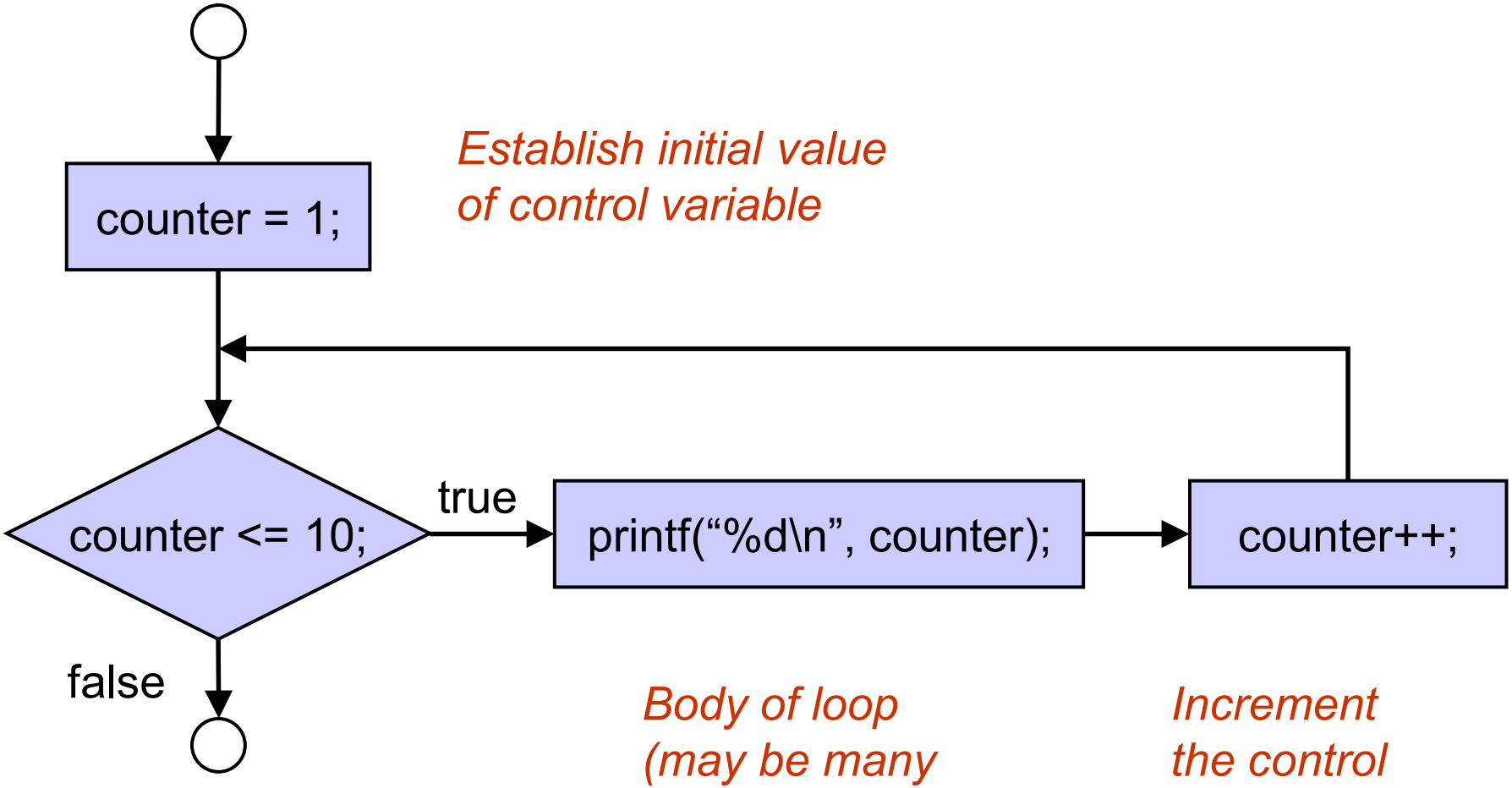
4.5 The `for` Statement : Notes III

Initialisation and incrementation

- "Increment" may be negative (decrement)
- If the loop-continuation condition is initially false
  - The body of the **`for`** statement is not performed
  - Control proceeds with the next statement after the **`for`** statement
- Control variable
  - Often printed or used inside **`for`** body, but not necessarily

```
int counter;
for( counter = 1; counter <= 10;  counter++) {
  printf( "%d\n", counter );
}
```

4.5 The **for** Statement : Notes IV



Establish initial value of control variable

Determine if final value has been reached

Body of loop (may be many statements)

Increment the control variable

4.6 for Examples I

for examples

- Vary the control variable from 1 to 100 in increments of 1

```
for ( i = 1; i <= 100; i++ )
```

- Vary the control variable from 100 to 1 in increments of -1

```
for ( i = 100; i >= 1; i-- )
```

- Vary the control variable from 7 to 77 in increments of 7

```
for ( i = 7; i <= 77; i += 7 )
```

- Vary the control variable from 20 to 2 in increments of -2

```
for ( i = 20; i >= 2; i -= 2 )
```

4.6 for Examples II

for examples (cont'd...)

- Vary the control variable from 1 to 100 in increments of 1

```
for ( i = 1; i < 101; i++ )
```

- Vary the control variable from 100 to 1 in increments of -1

```
for ( i = 100; i > 0; i-- )
```

- Vary the control variable over the following sequence: 2, 5, 8, 11, 14, 17.

```
for ( i = 2; i <= 17; i += 3 )
```

- Vary the control variable over the following sequence: 44, 33, 22, 11, 0.

```
for ( i = 44; i >= 0; i -= 11 )
```

4.6 for Design Example: Compound Interest I

for Problem statement

A person invests R1000.00 in a savings account yielding 5% interest annually. Assuming that all interest is left on deposit in the account, calculate and print the amount of money in the account at the end of each year for 10 years. Use the following formula for determining these amounts:

$$a = p(1 + r)^n,$$

where

- p is the original amount invested (the principal)
- r is the annual interest rate
- n is the number of years
- a is the amount on deposit at the end of the nth year

4.6 for Design Example: Compound Interest II

Pseudocode

Initialise principal to 1000
Initialise interest rate to 0.05

For years 1 to 10 in increments of 1
    Calculate investment amount at the end of the year
    Display the year and investment amount

4.6 for Design Example: Compound Interest III

C code

```c
/* Copied from Fig. 4.6 in Deitel&Deitel
 *  Calculating compound interest */
#include <stdio.h>
#include <math.h>

// Function main begins program execution
int main( void )
{
  double amount;                // amount on deposit
  double principal = 1000.0;    // starting principal
  double rate = 0.05;           // annual interest rate
  int year;                     // year counter

  printf( "%4s%21s\n", "Year", "Amount on deposit" ); // column head
```

4.6 for Design Example: Compound Interest IV

C code (cont'd...)

```
  for ( year = 1; year <= 10; year++ ) {
    // calculate investment amount for specified year
    amount = principal * pow( 1.0 + rate, year );
    printf( "%4d%21.2f\n", year, amount ); // output one table row
  } // end for

  return 0; // indicates program ended successfully
} // end main
```

4.6 for Design Example: Compound Interest V

Output

```
Year    Amount on deposit
   1              1050.00
   2              1102.50
   3              1157.63
   4              1215.51
   5              1276.28
   6              1340.10
   7              1407.10
   8              1477.46
   9              1551.33
  10              1628.89
```

Perspective

Today

Program Control I

- Review of repetition structures
- `for` statement
- `for` examples

Next lecture

Program Control II

- `do...while` statement

Homework

1. Study Sections 4.1-4.6 in Deitel & Deitel
2. Do Self Review Exercises 4.3, 4.4(a),(b)&(d) in Deitel & Deitel
3. Do Exercises 4.5(a),(d),(e)&(g), 4.7, 4.11, 4.16 in Deitel & Deitel