An Object-Oriented Approach to

Programming Logic and Design, 4rd Edition

Chapter 5 (1, 3, 7, 11)

Exercises

Exercise #1

a. Design the logic for a program that allows a user to enter 15 numbers, then displays them in the reverse order of their entry.

```
class Numbers

Main()

//Declarations

num numbers[15]

num i

output "Please enter 15 numbers."

for i = 0; i < 15; i = i + 1

        numbers[i] = input("Enter a number")

endfor

for i=14; i  >= 0; i = i – 1

        output number[i]

endfor

endClass
```

Exercise #7

Design the application logic for a company that wants a report containing a breakdown of payroll by department. Input includes each employee's department number, hourly salary, and number of hours worked. The output is a list of the seven departments in the company and the total gross payroll (rate times hours) for each department. The department names are shown in Table 5-4.

Department Number	Department Name
1	Personnel
2	Marketing
3	Manufacturing
4	Computer Services
5	Sales
6	Accounting
7	Shipping

class Payroll

Main()

//Declarations

str depts[7] = Personnel, Marketing, Manufacturing, ComputerServices, Sales, Accounting, Shipping

num totalGrossPay[7]

num enteredDept

num hours

num rate

num i

input(Please enter the department number. Enter 99 when you are finished.)

while enteredDepartment != 99

    input(Enter the hourly salary.)

input(Enter the hours worked.)

totalGrossPay[enteredDepartment – 1] = totalGrossPay[enteredDepartment – 1] + hours * rate

input(Please enter the department number. Enter 99 when you are finished.)

endwhile

for i = 0; i < 8; i = i + 1

output depts[i], totalGrossPay[i]

endfor

endClass

Chapter 5: Case Study #1

Case: Cost Is No Object

1. In earlier chapters, you developed programs for Cost Is No Object—a car rental service. Create an application that produces employee information for Cost Is No Object. The application prompts the user for an employee ID number, first and last names, street address, zip code, and job description code.

   Any time the user enters an invalid value, continue to reprompt the user for the same data. Invalid values are:

   - An employee ID number that is negative or greater than 999
   - A zip code that is not in the list of allowed zip codes
   - A job description code that is not between 10 and 19 inclusive

   Determine the employee's city and state based on the zip code, as described in Table 5-10. Determine the employee's job title based on the values in Table 5-11. Determine the employee's hourly pay rate based on the values in Table 5-12.

   When all the needed data has been entered correctly for an employee, output the ID number, first and last names, street address, city, state, zip code, job code, job title, and hourly pay rate.