Student Name: enter your name here
Class-Section: enter your section number here
Date: enter the creation date here
Plan Instructions V3

Plan for PA___ ←enter the PA number here

Why do you plan before you code?

NOTE: If you don't understand the context within which words are used in this document, look up the words in your textbook.

1. *Program Purpose*: What is the purpose of the program? To understand how to code a program you must first understand its purpose.

IPO CHART

Input	Processing (Calculations)	Output
What are the variables necessary to capture data into the program?	What type of processing will occur – particularly calculations?	What will the output look like? Looking at the output will tell you the logic of the program. *In addition to data, output will also have headers and labels. Headers identify the type of output and labels identify each specific data item in the output. Headers, labels, and spacing don't change so they are entered permanently in the program. Any output that is derived from variables (fields) is data that can change so indicate string and/or character data with Xs and numeric data with 9s and use Zs to indicate suppression of leading zeroes with $ signs and commas where needed.* *Example:* BOOK SALES FOR 9999 Emma's Bookmart DeZavala Road San Antonio, TX XXXXXXXXX Sales $ZZ,ZZ9.99 XXXXXXXXX Sales ZZ,ZZ9.99 TOTAL SALES: $ZZZ,ZZ9.99 PROFIT MARGIN: ZZ9.99%

Formatting Output: Be aware of the spacing, punctuation, and line advance requirements in the output. Double line advance means 1 blank space between lines, etc.

HEADER w/Variables

HEADERS w/Company Info

LINE LABELS & VARIABLES

SUMMARY LABELS & VARIABLES

2. *Class Diagram:*

Class Name	What is the name of the class program? Examples: OrderBooks, BookSales, RegisterClasses, BuyGroceries, *LastNameFirstInitialPANo*
Data Members or Variables	▪ Variables declared at the class level are *class variables or fields* and should be declared as private unless they belong to a superclass then they are declared as protected. ▪ Variables declared at the method level are local variables and have no access modifiers. ▪ If variables are to be used by more than one method, they should be class variables or fields.
Class Data Members	▪ **Modifiers or access modifiers:** *designate how accessible a variable is and are used in declaring variables at the class level. Class variables are declared the same way as ones for methods, except they require a **modifier** and the modifier appears before the variable name.* ○ - sign means private *(belongs only to the class)* ○ + sign means public *(belongs to any class)* ○ # sign means protected *(belongs to classes in the same family)* ○ **Format:** *modifier variableName: dataType* ○ Multiple variables of the same type can be listed together: ○ Ex: -choice, quantity, bookCount, size, count: int -keyboard: Scanner -price, totalCost, taxAmount, subtotal, averageBookPrice: double -totals: double[] -itemName, salesReceipt, receiptHeader: String -decimal: DecimalFormat OR -decimal: DecimalFormat("#,##0.00") -money: NumberFormat -SALES_TAX: double
Method Data Members	▪ **Variables for input and for output.** Most of the time output variables are input variables. Sometimes there are variables that contain data that has been processed such as variables involved in calculations. Identifiers are the names you give to the variables. Method or local variables do not need modifiers.

	▪ **Format:** *identifierName: dataType* ○ Ex: noOfBooks: int totalBkSales: double ▪ **Reference variables for objects. These variables reference or point to objects of programs that contain multiple values.** ▪ **Format:** *objectName: dataType* ○ Ex: input: Scanner ○ Ex: totals[]: double ▪ **If listing more than one method then separate variables for each method by the method name.** Ex: main() noOfBooks, month, year: int avgBkPrice, totalBkSales, bookSales, discountMargin, profitMargin: double monthName: String input: Scanner displayData() now: Date date: String formatter: SimpleDateFormat
Methods	▪ An application is a class program that usually has one method and it is the main(). The main() puts into motion the rest of the code. ▪ A class is a program without the main() and it usually has 3 basic methods ○ One that captures input. ○ One that processes that input. ○ One that takes care of the output. ○ All other methods support the above three. ○ If code is being repeated in each of the basic methods, then it is a good idea to put the code in a method of its own so the method can be called by the basic methods. ○ All methods are declared with a modifier. ▪ **Format:** *modifier methodName(parameter list): returnType* ○ Ex: +main(String[] args): void *A public method that receives values in a String array through its parameter list and returns nothing* ○ Ex: -displayData(): void *A private method that receives nothing through its parameter list and returns nothing*

- Ex: #getAge(age: int): void *A protected method that receives an integer value through the parameter variable age and returns nothing*
- Ex: +getNoBooks(): int *A public method that receives nothing through its parameter list and returns an int value to a calling method or statement*

Ex: +calcSale(qty: int, price: double): double

The above is a public method that receives an integer argument and a double argument through its parameter list and returns a value of type double to the calling method or statement. NOTE: Parameter variables are scoped (only belong) to the method that declares them in its parameter list.

- More examples:
  +pickBooks(): void
  +processSales(): void
  +getItemName(): String
  +bookCosts(noBooks: int): void
  +getTotals(): double[]
  +sumTotals(monthlyTotals: double[]): double

3. *Program Logic:*

import Stmts: *List the classes to be imported so your program can use their code.*
import Scanner

Class Header: *What is the name of the class?*
public class BookSales

Class Variables (Fields): *Refer to the Class Data Members section of the class diagram for a list of class-level variables or fields which should always be private (-).*
N/A *If no class variables; otherwise list them as:* ***modifier dataType variableName*** ***Ex:*** **private String employeeName**

Method Purpose: *What is the purpose of the method?*	
Method Header	*What is the method name?*
Method Variables	*Refer to the Class Diagram for what they are.*
CODE	

Prompts	*Identify the variables requiring prompts. Prompt according to the program specifications in the assignment instructions.*
Print	*Identify the variables to be printed and print according to output specifications in the IPO chart.*

Algorithms	○ *The control structures will be pseudocoded here in logical sequence.*
	○ *The control structures such as the loops and decisions are pseudocoded similar to the programming language except you don't use braces and semicolons*
	○ *Some test expressions are pseudocoded using language independent symbols*
	
	○ *Ex: **if** and **if-else** decision structures*
	
	`if choice >= 1`
	`  statement(s)`
	`endIf` *(a close brace in Java)*
	
	`if choice < 1`
	`  statement(s)`
	`endIf`
	
	`if choice = 1` *(NOTE: equality as Java code is ==)*
	`  statement(s)`
	`else`
	`  statement(s)`
	`endIf` *(a close brace in Java for the else)*
	
	`if choice > 0 AND choice < 5` *(NOTE: AND as code is &&)*
	`  statement(s)`
	`endIf`
	`if choice <= 0 OR choice > 4` *(NOTE: OR as code is ||)*
	`  statement(s)`
	`endIf`
	
	○ *Ex: a **switch** decision structure evaluates an int, byte, short or char variable or an expression that produces a result of those data types. Break statements are not necessary in the pseudocode as they are implied, but they have to be coded in the real program.*
	
	`switch on` *choice ← where* `choice` *is the variable*
	`  case 1:  statement(s)`
	`  case 2:  statement(s)`
	`  default:  statement(s)`
	`endSwitch` *(a close brace in Java)*
	
	○ *Ex: looping structures are the do-while, while, for*

```
do
    statements

    statement that will eventually make the test
    expression false so the loop ends

while test expression

  Example of test expression:

  toContinue = 'y' OR toContinue = 'Y'
```

↗ *The above relational conditions joined by the logical operator OR is an example of a test expression when true will cause re-entry into the do-while loop; hence, post-test loop.*

```
while test expression   ←The while loop will only be entered when
                          the test expression is true; hence, a pre-test loop.
  statements
  statement that will eventually make the test
  expression false so the loop ends

endWhile (a close brace in Java)
```

The **for** *is also a pre-test loop. "until" means up to but not including whereas*

🡻 *"thru" means including*

```
for(i from 0 until/thru count)

  statements

endFor (a close brace in Java)
```

- *Ex: concatenation statements in pseudocode*

```
receiptHeader = "Wal-Mart"
                "DeZavala Road"
                "San Antonio, TX"

salesReceipt += receiptHeader
                shoppingCart

                "        SUBTOTAL:  ", $, subtotal

                "     TAX @ 8.25%:  ",    taxAmount

                "           TOTAL:  ", $, totalCost
```

ACTUAL CODE:

Using the money object to call the format() method from the NumberFormat class by sending it a value from the variable totalCost.	salesReceipt += String.format(receiptHeader + shoppingCart + "%n SUBTOTAL: " + money.format(subtotal) + "%n TAX @ 8.25%: " + decimal.format(taxAmount) +"%n TOTAL: " + money.format(totalCost) + "%n"); salesReceipt += String.format(receiptHeader + shoppingCart + "%n SUBTOTAL: $%,.2f" + "%n TAX @ 8.25%: %,.2f" + "%n TOTAL: $%,.2f%n", subtotal, taxAmount, totalCost);

EXAMPLE:

1. *Program Purpose*: This program processes the monthly and total book sales (yearly) for Emma's Bookmart for a given year. The discount margin is .5% if total book sales is greater than 17000.00. The profit margin is 10% if total sales is greater than 50000.00; otherwise, it is 7.5%.

Input	Processing (Calculations)	Output
year noOfBooks avgBkPrice	bookSales = noOfBooks * avgBkPrice totalBkSales = totalBkSales + bookSales	BOOK SALES FOR 9999 Emma's Bookmart DeZavala Road San Antonio, TX XXXXXXXXX Sales $ZZ,ZZ9.99 XXXXXXXXX Sales ZZ,ZZ9.99 TOTAL SALES: $ZZZ,ZZ9.99 PROFIT MARGIN: ZZ9.99%

2. *Class Diagram:*

Class Name | BookSales

Class Data Members	N/A
Method Data Members	main() noOfBooks, month, year: int avgBkPrice, totalBkSales, bookSales, discountMargin, profitMargin: double monthName: String input: Scanner
Methods	+main(String[] args): static void

import Stmts: ***import NameOfClass***
import Scanner

Class Header: ***public class NameOfYourClass***
public class BookSales

Class Variables (Fields): *Refer to the Class Data Members section of the class diagram for a list of class variables which should always be private (-).*
N/A

Method Purpose: ***The main() prompts for the year of the book sales, the number of books sold for a given month, and the average book price for that month. It calculates the revenue (bookSales) for each month, the total book sales for the year, determines the discount margin and profit margin, prints the monthly sales, the total sales, and the profit margin.***

Method Header	public static void main(String[ ] args)
Method Variables	int noOfBooks, month = 1, year double avgBkPrice, totalBkSales, bookSales, discountMargin, profitMargin String monthName Scanner input

CODE

Prompts	Input Variables	Input Prompt
1	year	"What is the year? "
2	noOfBooks	"How many books were sold in ", monthName, "? "
3	avgBkPrice	"What is the average price of the books? "

Print	**Output** *The output is not prompts. Output can be messages, final results ordered according to when they appear in the program.*
1	"BOOK SALES FOR ", year "Emma's Bookmart" "DeZavala Road" "San Antonio, TX" monthName, " Sales $", bookSales monthName, " Sales ", bookSales "TOTAL SALES: $", totalBkSales "PROFIT MARGIN: ", profitMargin, "%"
2	*Additional output other than the final output such as error message, thank you message.*
3	*Additional output other than the final output such as error message, thank you message.*
Algorithms	Prompt 1 while month > 0 AND month < 13 switch on month case 1: monthName = "January" case 2: monthName = "February" case 3: monthName = "March" : . default: monthName = "Invalid" endSwitch Prompts 2, 3 bookSales = noOfBooks * avgBkPrice totalBkSales = totalBkSales + bookSales month = month + 1 endWhile **OR** do switch on month case 1: monthName = "January" case 2: monthName = "February" case 3: monthName = "March" : . default: monthName = "Invalid" endSwitch

```
     Prompts 2, 3

     bookSales = noOfBooks * avgBkPrice
     totalBkSales = totalBkSales + bookSales

     month = month + 1
while month > 0 AND month < 13

OR

for(month from 1 thru 12)

     switch on month
          case 1:  monthName = "January"
          case 2:  monthName = "February"
          case 3:  monthName = "March"
          :
          .
          default:  monthName = "Invalid"
     endSwitch

     Prompts 2, 3

     bookSales = noOfBooks * avgBkPrice
     totalBkSales = totalBkSales + bookSales

endFor

if totalBkSales > 17000.00
     discountMargin = .05
endIf

if totalBkSales > 50000.00
     profitMargin = .10
else
     profitMargin = .075
endIf

Print 1

Stop
```


Comments:

1. **Rules for Declaring Variables:** Results in camel casing.
   a. **For Variables (Fields) and Methods:** Variable or method names begin with the first letter in lowercase and the first letter in each successive word in uppercase. This is why when pseudocoding in Word you should disable the automatic

capitalization of first letters. Go to **AutoCorrec**t under **Office Button**, **Word Options**, then **Proofing**, then **AutoCorrect**.

b. **For Programs:** Program names begin with the first letter in uppercase and the first letter in each successive word in uppercase.