**CONFIDENTIAL**



## UNIVERSITI TEKNOLOGI MALAYSIA

# **MID-TERM TEST**

# **SEMESTER II 2015/2016**

|  |  |
| --- | --- |
| **SUBJECT CODE** | **: SCSJ10123** |
| **SUBJECT NAME** | : PROGRAMMING TECHNIQUE II |
| **YEAR/COURSE** | **: 1 (SCSJ / SCSV / SCSB / SCSR)** |
| **TIME** | **: 2.00 p.m. – 5.00 p.m.** |
| **DATE** | **: 12 APRIL 2015** |
| **VENUE** | **: N28 BK1-BK7** |

**INSTRUCTIONS TO THE STUDENTS:**

This test book consists of :

PART A: (6 Structured Questions) 65

PART B: (1 Programming Question) 35

TOTAL 100

**ANSWER ALL QUESTIONS IN THE ANSWER BOOKLET PROVIDED.**

|  |  |
| --- | --- |
| **Name** |  |
| **I/C No.** |  |
| **Year/Course** |  |
| **Section** |  |
| **Lecturer’s Name** |  |

*(This question booklet consists of 17 pages including this page.)*

**PART A: STRUCTURED QUESTIONS TOTAL – 65 MARKS**

**QUESTION 1 [EN ASHARI] [10 marks]**

Program 1 is intended to read the data of a list of employees from the keyboard and create a file containing the following records: Name, Identity Card (IC) number, state name and monthly income. Then, the program reads back the file for further processing and finally prints the results onto the screen. As for the state name, the user needs to enter the code and the program will determine the name based on the following conversion.

State code State

1. Johor
2. Kedah
3. Kelantan
4. Melaka
5. Negri Sembilan

Program 1 will request the user to:

* Key in the number of employees. Read each employee’s data from the keyboard and write them to a file. Use only **one file** for BOTH input and output data.
* Read data from the file and do the following:

1. Determine the exact age (day, month and year) of each employee and display his/her name, day, month and year of birth on the screen. Day, month and year are determined by the first six (6) digits of the IC. (Assume that the year considered for this case is between 1900-1999).
2. Find employees with income less than RM5000 and display their names and state name on the screen.

Complete the program based on the questions ***(a) – (s)*** stated as commentaries in the given code below.

|  |  |
| --- | --- |
| **1**  **2**  **3**  **4**  **5**  **6**  **7**  **8**  **9**  **10**  **11**  **12**  **13**  **14**  **15**  **16**  **17**  **18**  **19**  **20**  **21**  **22**  **23**  **24**  **25**  **26**  **27**  **28**  **29**  **30**  **31**  **32**  **33**  **34**  **35**  **36**  **37**  **38**  **39**  **40**  **41**  **42**  **43**  **44**  **45**  **46**  **47**  **48**  **49**  **50**  **51**  **52**  **53**  **54**  **55**  **56**  **57**  **58**  **59**  **60**  **61**  **62**  **63**  **64**  **65**  **66**  **67**  **68**  **69**  **70**  **71**  **72**  **73**  **74**  **75**  **76**  **77**  **78**  **79**  **80**  **81**  **82**  **83**  **84**  **85**  **86**  **87**  **88**  **89**  **90**  **91**  **92**  **93**  **94**  **95**  **96**  **97**  **98**  **99** | **// Program 1**  **#include <iostream>**  **#\_\_\_\_\_\_\_\_\_\_\_\_\_\_ // *a)* include the library for file operations [0.5m]**  **#\_\_\_\_\_\_\_\_\_\_\_\_\_\_ // *b)* include the library for c-strings [0.5m]**  **#\_\_\_\_\_\_\_\_\_\_\_\_\_\_ // *c)* include the C standard library [0.5m]**  **using namespace std;**  **int main()**  **{**  **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ // *d)* declare c-string variables**  **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ // name, ic & stateName**  **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ // with appropriate size. [1.5m]**  **int stateCode**    **double monthlyIncome = 0.0;**  **int numOfEmployee;**  **\_\_\_\_\_\_\_\_\_\_\_\_\_ // *e)* Declare the file variable for output file [0.5m]**  **\_\_\_\_\_\_\_\_\_\_\_\_\_ // *f)* Open output file named “outfile.txt” [0.5m]**    **if (\_\_\_\_\_\_\_\_)// *g)* check if there is an error in opening file [0.5m]**  **{ cout << "Cannot open file" << endl;**  **exit(1); }**  **else**  **{ cout << "How many employees?: ";**  **cin >> numOfEmployee; // Ask the user for the number of employee**  **// *h)* Write the loop to read data for all employees [0.5m]**  **for (\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_)**  **{ cin.ignore();**  **cout << "Enter name: " ;**  **// *i)* read the employee’s name [0.5m]**  **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**    **cout << "IC: ";**  **cin.ignore();**  **// *j)* read the employee’s IC number [0.5m]**  **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**    **cout << "State Code: ";**  **cin >> stateCode;**    **if (stateCode == 1) strcpy(stateName,"Johor");**  **else if (stateCode == 2)strcpy(stateName, "Kedah");**  **else if (stateCode == 3)strcpy(stateName, "Kelantan");**  **else if (stateCode == 4)strcpy(stateName, "Melaka");**  **else if (stateCode == 5)strcpy(stateName, "Negeri Sembilan");**    **cout << "Monthly Income: ";**  **cin >> monthlyIncome;**    **out << name << endl << ic << endl << stateName << endl**  **<< monthlyIncome << endl;**  **}**  **}**  **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ // *k)* close the output file [0.5m]**  **// *l)* Re-open the same file (i.e. “outfile.txt”) but this time**  **// as an input file [0.5m]**  **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**  **char year[2], month[2], day[2];**  **//*m)* check whether there is still data to read in [0.5m]**  **while \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_)**  **{**  **// *n)* read the employee’s IC number from the file [0.5m]**  **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**    **// *o)* read stateName from the file [0.5m]**  **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**    **// *p)* read monthlyIncome from the file [0.5m]**  **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**  **cout << "name: " << name << " ic: " << ic << " State:**  **" << stateName << " income: " << monthlyIncome << endl;**    **// *q)* Read 2 digits from IC for day [0.5m]**  **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**  **// *r)* Read 2 digits from IC for month [0.5m]**  **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**  **// *s)* Read 2 digits from IC for Year [0.5m]**  **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**    **cout << "Date of birth: " << day << " of " << month <<**  **"19" << year << endl;**  **if (monthlyIncome < 5000) {**  **cout << "Income less than RM5000" << endl;**  **cout << " Name: " << name << " State: " <<**  **stateName << endl;**  **}**  **in.getline (name, 31); }**  **}**  **in.close(); // close input file**  **return 0; }** |

**QUESTION 2 [EN.JUMAIL] [12 marks]**

Program 2 is meant to read a list of student records from a text file and copy the list into binary files. A student record consists of the student’s name, matriculation number, and course. Each record in the text file is stored in a line in which all the fields are separated by asterisks (**\***). The first field is the student’s name, second is the matriculation number and third is the course. Figure 1 illustrates an example of the text file.

|  |
| --- |
| **NUR HIDAYAH ABDUL HALIM\*BC12345\*Bachelor of Computer Science**  **SITI AMIRAH ZULKIFLI\*BA56789\*Bachelor of Architecture**  **NUR AMALEENA HARUN\*BE31264\*Bachelor of Electrical Engineering**  **RAZALI ABU BAKAR\*DC43783\*Diploma of Building**  **OTHMAN KAMARUL\*BE23445\*Bachelor of Electrical Engineering**  **SITI NURHAZILA HUSSAIN\*BC54345\*Bachelor of Computer Science**  **NUR FATIHA KAMARUDDIN\*DA23908\*Diploma of Architecture**  **MUHAMMAD SHAHRUL NIZAM FAZLAN\*BC23466\*Bachelor of Civil Engineering**  **RINA ABDUL SALAM\*MH36789\*Master of Humanity and Social Science**  **HAMID JALAL\*BE72323\*Bachelor of Electrical Engineering**  **NUR HAKIMAH MOHAMAD ZULHISYAM\*BA23489\*Bachelor of Architecture** |

**Figure 1:** An example of input file

Complete the program based on questions **(a)** - **(g)** stated as commentaries in the given code .

|  |  |
| --- | --- |
| **1**  **2**  **3**  **4**  **5**  **6**  **7**  **8**  **9**  **10**  **11**  **12**  **13**  **14**  **15**  **16**  **17**  **18**  **19**  **20**  **21**  **22**  **23**  **24**  **25**  **26**  **27**  **28**  **29**  **30**  **31**  **32**  **33**  **34**  **35**  **36**  **37**  **38**  **39**  **40**  **41**  **42**  **43**  **44**  **45**  **46**  **47**  **48**  **49**  **50**  **51**  **52**  **53**  **54**  **55**  **56**  **57**  **58**  **59**  **60**  **61**  **62**  **63**  **64**  **65**  **66**  **67**  **68**  **69**  **70**  **71**  **72**  **73**  **74**  **75**  **76**  **77**  **78**  **79**  **80**  **81**  **82**  **83**  **84**  **85**  **86**  **87**  **88**  **89**  **90**  **91**  **92**  **93**  **94**  **95**  **96**  **97**  **98**  **99**  **100** | **// Program 2**  **#include <iostream>**  **#include <fstream>**  **using namespace std;**  **struct Student**  **{ char name[50];**  **char matric[11];**  **char course[50];**  **};**  **void readStudent(fstream&, Student&);**  **void openOutBinFile(const char[], fstream&);**  **void writeStudent(fstream&, Student);**  **int main()**  **{**  **Student studList[50];** ***// To hold a list of student records***  **int count;** ***// The number of students***    **fstream inputFile;**  **fstream outputFile1;**  **fstream outputFile2;**    **inputFile.open("students.txt", ios::in);**    ***//(a). Test whether opening the text file succeeded***  **if (\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_){ // 0.5m**  **cout << "Unable to open the input file" << endl;**  **return 0;**  **}**  ***// (b). Read all the student records from the text file into the***  ***// array studList. Use the function readStudent to read a student***  ***// record.***  **count = 0;**  **while (\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_){ // 0.5m**  **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ // 1m**  **count = count + 1;**  **}**    ***// (c). Using the function openBinFile, open two binary files for***  ***// writing, outputFile1 and outputFile2, respectively.***  **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ // 1m \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ // 1m**  ***// (d). Using a loop and the function writeStudent ,save the***  ***// records of studList into the first output file***    **for (\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_) // 1m**  **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ // 1m**  ***// (e). Save the same list into the second output file by writing***  ***// all records at once.***  **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**  **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ // 2m**    **inputFile.close();**  **outputFile1.close();**  **outputFile2.close();**  **return 0;**  **}**  ***// (f). Complete the definition of function readStudent.***  ***// This function reads a student record from a text file.***  **void readStudent(fstream &txtFile, Student &stud)**  **{**  **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ // 1m**  **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ // 1m**  **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ // 1m**  **}**  ***// (g). Complete the definition of function openOutBinFile.***  ***// This function opens a binary file for writing (output)***  **void openOutBinFile(const char fileName[], fstream &binFile)**  **{**  **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ // 1m**  **}**  ***// The function writeStudent saves a student record into a binary file***  **void writeStudent(fstream &binFile, Student stud)**  **{**  **binFile.write(reinterpret\_cast<char\*>(&stud),**  **sizeof(Student) );**  **}** |

**QUESTION 3 [REUSED - kiv] [7 marks]**

Consider the following definitions and initializations. Assume that the address of **chr1** is 0x7f5aab and the address of **chr2** is 0x7f5bbc.

Draw the memory layout for the following program segments.

**char chr1 = '%', chr2 = '$';**

**char \*pt1 = &chr1;**

**char \*pt2 = &chr2;**

**char \*pt3;**

(2 marks)

**pt3 = &chr2;**

**cout << "\*pt3 = " << \*pt3 << endl;**

(1 marks)

**pt3 = pt1;**

**cout << "\*pt3 = " << \*pt3 << ", pt3 = " << pt3 << endl;**

(1 marks)

**\*pt1 = \*pt2;**

**cout << "\*pt1 = " << \*pt1 << ", pt1 = " << pt1 <<endl;**

(1 marks)

What will be printed when the statements above executed sequentially?

(2 marks)

**QUESTION 4 [DR.SARINA] [10 marks]**

4 (a) The main function of a program is given as in **Program 3a** shown below. The program counts for salary data through the function **getSalary** and calculates the total salary.

|  |  |
| --- | --- |
| **1**  **2**  **3**  **4**  **5**  **6**  **7**  **8**  **9**  **10**  **11**  **12**  **13**  **14** | **// Program 3a**  **int main()**  **{**  **double hours = getWeeklyHours();**  **double salary = \*getSalary();**  **cout << "Weekly Hours: " << hours << endl;**  **cout << "Hourly Salary: " << salary << endl;**  **double weeklySalary = hours \* salary;**  **cout << "Weekly Salary: " << weeklySalary << endl;**  **return 0;**  **}** |

The output for the program is as below.

Weekly Hours: 46.5

Hourly Salary: 26.48

Weekly Salary: 1231.32

(i) Write the code to define a **getSalary** function to return a pointer. In this function, a variable **salary** is assigned to **26.48** and it holds the address that points to the **hourlySalary**. *Note*: you should use pointer notation in this function.

(4 marks)

(ii) Write the code that defines the **getWeeklyHours** function to return an address to the local variable in the function. The local variable, **n** with the assigned value **46.50** is passed to the address of **hours**. *Note*: you should use address as argument to the function. (4 marks)

4. (b) What is the output of **Program 3b** below:

|  |  |
| --- | --- |
| **1**  **2**  **3**  **4**  **5**  **6**  **7**  **8**  **9**  **10**  **11**  **12**  **13**  **14**  **15**  **16**  **17**  **18**  **19**  **20**  **21**  **22**  **23**  **24**  **25**  **26** | **// Program 3b**  **#include <iostream>**  **#include <iomanip>**  **using namespace std;**  **struct ShoeType**  **{ string style;**  **double price; };**  **int main()**  **{ ShoeType shoe1, shoe2, \*s1, \*s2;**  **s1 = &shoe1;**  **s2 = &shoe2;**    **cout << fixed << setprecision(2);**  **shoe1.style = "Sandal";**  **shoe1.price = 98.98;**  **cout << s1->style << "\t" << " RM" << s1->price << endl;**  **\*s2 = \*s1;**  **//Put shoe2 on sale!**  **s2->style = "Slipper";**  **s2->price = (\*s1).price\*50/100;**  **cout << shoe2.style <<"\t"<< " RM" << shoe2.price << endl;**    **return 0; }** |

(2 marks)

**QUESTION 5 [PM.NORANIAH] [6 marks]**

5. **Program 4** demonstrates the use of dynamic arrays. Complete the program based on the questions **(a)** - **(e)** as stated in the comments provided in the given code below:

|  |  |
| --- | --- |
| **1**  **2**  **3**  **4**  **5**  **6**  **7**  **8**  **9**  **10**  **11**  **12**  **13**  **14**  **15**  **16**  **17**  **18**  **19**  **20**  **21**  **22**  **23**  **24**  **25**  **26**  **27**  **28**  **29**  **30**  **31**  **32**  **33**  **34**  **35**  **36**  **37**  **38**  **39**  **40**  **41**  **42**  **43**  **44**  **45**  **46** | **// Program 4**  **#include <iostream>**  **#include <iomanip>**  **using namespace std;**  **int main()**  **{**  **float \*monthSales; // a pointer used to point to an array**  **// holding monthly sales**  **float total = 0; // total of all sales**  **int numOfSales; // number of sales to be processed**  **int count; // loop counter**  **cout << fixed << showpoint << setprecision(2);**  **cout << "How many monthly sales will be processed? ";**  **cin >> numOfSales;**  **//(a) To allocate memory for the array pointed to by monthSales.**  **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ // [2m]**  **//(b)Fill in condition to determine if memory has beenallocated**  **if ( \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ ) // [1m]**  **{**  **cout << "Error allocating memory!\n";**  **return 1; }**  **cout << "Enter the sales below\n";**  **for ( count = 0; count < numOfSales; count++)**  **{**  **cout << "Sales for Month number "**  **//(c) Fill in code to show the number of the month**  **<< \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ // [1m]**  **//(d) Fill in code to bring sales into an element of the array**  **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ // [1m] }**  **for ( count = 0; count < numOfSales; count++)**  **total = total + monthSales[count];**  **//(e) Fill in the code to deallocate memory assigned to the array.**  **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ // [1m]**  **return 0; }** |

**QUESTION 6 [20 marks]**

For a quadratic equation  2 + + c = 0 (where a, b and c are coefficients), its roots is given by following the formula :

and

The program will ask the user to enter coefficients a, b and c and computes the roots of a quadratic equation. The computation should follow the following **class** specification:

**private** members :

**a, b, c** type double coefficients values

**discriminant** type double determinant value

**calcDiscriminant()** function with return inline function member to calculate the determinant value,

**public** members :

**Compute(double, double, double)** a constructor that takes arguments 3 arguments to initialise the coefficients value, and invoke **calcDiscriminant()** to obtain the determinant value

**positive\_root()** function to calculate and return value

**negative\_root()** function to calculate and return value

a) Define a class **Compute** based on the specifications given above.

(5 marks)

b) Write the definition of the constructor **Compute(double, double, double)** to initialize the private member variable **a, b, c** to the value of the parameter, and assign the private member variable **discriminant** to invoke **calcDiscriminant()**function.

(4 marks)

c) Based on the roots of quadratic equation given above, define two function members for **positive\_root()** and **negative\_root()** to calculate and values.

(4 marks)

d) The following **main()** function of **Program 5** is incomplete. Fill in the three missing statements according to the instructions as commented in (a) – (c).

(3 marks)

|  |  |
| --- | --- |
| **1**  **2**  **3**  **4**  **5**  **6**  **7**  **8**  **9**  **10**  **11**  **12**  **13**  **14**  **15**  **16**  **17**  **18**  **19** | **// Program 5**  **int main()**  **{**  **double a, b, c;**  **cout << "Enter coefficients a, b and c: ";**  **cin >> a >> b >> c;**  **// (a)Pass arguments a, b, c to the constructor**  **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_;**  **cout << "The roots of the quadratic equation are x = ";**  **// (b)Execute function members to display the value x1**  **cout << \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_;**  **cout << ", ";**  **// (c)Execute function members to display the value x2**  **cout << \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ << endl;**    **return 0; }** |

e) The **main()** function is completed and you have compiled them without errors.

1. What is the output of the program if the inputs entered by the user is **a = 4, b = 5** and **c = 1** (2 marks)

1. Based on the result you found in e (i), what is the value of **discriminant**? Assume the condition given in Pseudocode 1 is working in the program, what is the output that will be printed by the program? (2 marks)

Pseudocode 1 : If (discriminant = 0)

Print “The equation has a single root.”

Else if (discriminant < 0)

Print “The equation has two complex roots.”

Else

Print “The equation has two real roots”

**PART B: PROGRAMMING QUESTION TOTAL – 35 MARKS**

Write a complete C++ program to calculate total daily sales for Cutie Bakery. The program should have a class named **Bread**. The **Bread** class has the following **private** member variables:

* **code** - is of type **string** that holds the code of the bread. The code representing the name of the bread based on Table 1 and the price of the bread. For example the **101250** represents that the bread’s
  + name is Raisin Croissant (i.e., taken from the first three characters)
  + price is RM 2.50 (i.e., taken from the last three characters)

**Table 1:** Bread id and bread name

|  |  |
| --- | --- |
| **Bread Id** | **Bread Name** |
| 101 | Raisin Croissant |
| 102 | Sugar Croissant |
| 201 | Sambal Bun |
| 202 | Cheese Bun |
| Other than the above | Kaya Braid |

* **price** - is of type **float** that holds the price of the bread.
* **quantity** - is of type **int** that holds the quantity of the bread sold.
* **name** - is of type **string** that holds the name of the bread.

The class should also have the following **public** member functions (methods):

* The inline mutator functions for all member variables.
* Two inline accessor functions, **getCode** and **getQuantity** that return the values from private member variables, **code** and **quantity**.
* **getPrice** is an accessor function that returns the value of **private** member variable, **price**. The bread’s price is taken from the last three characters from bread’s code. The function should define outside the class declaration.
* **getName** is an accessor function that returns the value of **private** member variable, **name**. The bread’s name is known from the first three characters and from bread’s code based on bread’s id and name in the Table 1. The function should define outside the class declaration.

The program should have a standalone function named **calculateTotal** that received an array of object from class **Bread**. The function is used to calculate and returns the total daily sales for Cutie Bakery. In the **main** function of the program:

* Create an array of four (4) objects from class **Bread**.
* Prompt the user to enter the code of bread and the quantity of breads sold.
* Set the member variables (**code** and **quantity**) of each object in the array using the mutator functions. Data should be read from the keyboard. You must use a loop for doing this.
* Print the bread information in tabular form. Your program should use another loop for doing this.
* Calculate and print the total daily sales of the bread.

Figure 2 is an example run of the program. Note that the text in bold indicates input entered by the user.

Please enter the information of bread:

Bread #1

Code: **101350**

Quantity: **6**

Bread #2

Code: **102280**

Quantity: **5**

Bread #3

Code: **201320**

Quantity: **4**

Bread #4

Code: **202430**

Quantity: **6**

Record of Daily Bread Sales

No Bread Code Bread Name Price(RM) Quantity

1 101350 Raisin Croissant 3.50 6

2 102280 Sugar Croissant 2.80 5

3 201320 Sambal Bun 3.20 4

4 202430 Cheese Bun 4.30 6

Total daily sales: RM73.60

Press any key to continue . . .

**Figure 2:** Example run