

ScreeIO.txt

```
1 *****
2 * PROGRAMMED BY : Saul Moreno
3 * STUDENT ID    : 269491
4 * CLASS         : CS 1B MW-7:30pm
5 * ASSIGNMENT #5 : Enhanced Employees
6 *****
7
8 This Program will display the Data for the C1SCE Employee, The Programmers
9 and Software Architects. It will display all other pertinent data
10
11 Data:
12 C1SCEmployees
13
14 These three examples are me just setting the employee's information automatically
15 Name           Employee's Id Phone      Age  Gender Job Title      Salary      Date
16 Joe Calculus    64879          949-555-1234  42   M      Math Wiz          $110000     8/31/2017
17 Mary Algebra    76309          213-555-5555  22   F      Math Wiz          $170123     5/08/2017
18 Jo Trig        12345          714-703-1234  29   F      Math Contact      $200000     12/25/2016
19
20 Address         City           State Zip Code
21 1234 Main Avenue Laguna Niguel  CA   92677
22 3333 Marguerite Pwky Mission Viejo  CA   92646
23 9876 Elm Street  San Clemente   CA   92672
```

main.cpp

```
1 /*****
2  * AUTHOR      : Saul Moreno
3  * STUDENT ID  : 269491
4  * ASSIGNMENT# : 4
5  * CLASS       : CS1C
6  * SECTION    : MW 5:00pm
7  * DUE DATE    : 3/27/2020
8  *****/
9
10 #include "header.h"
11 #include "date.h"
12 #include "employee.h"
13 #include "programmer.h"
14 #include "softwareArchitect.h"
15 #include "softwareTester.h"
16
17 int main()
18 {
19     Employee firstEmployee;    // variable that has access to the class
20     Programmer programOne;     // variable that has access to the class
21     SoftwareArchitect softOne; // variable that has access to the class
22     SoftwareTester *tPtr;      // pointer that points to the class
23     SoftwareTester lived;
24     tPtr = new SoftwareTester;
25
26     //Calls the functions to print out class header
27     PrintHeader("Enhanced Employees", 4, 'A');
28
29     cout << "This Program will display the Data for the C1SCE Employee, The"
30          << " Programmers" << endl
31          << "and Software Architects. It will display all other pertinent data";
32     cout << endl << endl;
33
34     cout << "Data:" << endl << "C1SCEmployees\n\n";
35
36     cout << "These three examples are me just setting the employee's information"
37          << " automatically\n";
38     //Calls function to display the table
39     firstEmployee.DisplayTable();
40     //Calls the function to set the initial values
41     firstEmployee.SetInitial("Joe Calculus", 64879, "949-555-1234", 42, 'M',
42                             "Math Wiz", 110000, "8/31/2017");
43     //Calls the functions to display the employee's information
44     firstEmployee.Display();
45     cout << endl;
46     //Calls the function to set the initial values
47     firstEmployee.SetInitial("Mary Algebra", 76309, "213-555-5555", 22, 'F',
48                             "Math Wiz", 170123, "5/08/2017");
49     //Calls the functions to display the employee's information
50     firstEmployee.Display();
51     cout << endl;
52     //Calls the function to set the initial values
53     firstEmployee.SetInitial("Jo Trig", 12345, "714-703-1234", 29, 'F',
54                             "Math Contact", 200000, "12/25/2016");
55     //Calls the functions to display the employee's information
56     firstEmployee.Display();
57     cout << endl << endl;
```

main.cpp

```
58 //Call the function to display the table top
59 tPtr->Display();
60 cout << endl;
61 //Call the function to set the employee's address
62 lived.Address(tPtr, "1234 Main Avenue", "Laguna Niguel", "CA",
63                 "92677");
64 cout << endl;
65 //Call the function to set the employee's address
66 lived.Address(tPtr, "3333 Marguerite Pwky", "Mission Viejo", "CA",
67                 "92646");
68 cout << endl;
69 //Call the function to set the employee's address
70 lived.Address(tPtr, "9876 Elm Street", "San Clemente",
71                 "CA", "92672");
72
73 return 0;
74 }
75
```

header.h

```
1 /*****
2  * AUTHOR      : Saul Moreno
3  * STUDENT ID  : 269491
4  * ASSIGNMENT# : 4
5  * CLASS       : CS1C
6  * SECTION    : MW 5:00pm
7  * DUE DATE    : 3/27/2020
8  *****/
9
10 #ifndef HEADER_H_
11 #define HEADER_H_
12
13 #include <iostream> // input and output
14 #include <iomanip>   // setprecision and setw
15 #include <string>    // allows to use strings
16 #include <limits>    //
17 #include <ios>       //
18 #include <fstream>   // file in & out
19 #include <time.h>    // system time
20 #include <stdlib.h>  // srand and rand
21 using namespace std;
22
23 void PrintHeader(string asName, int asNum, char asType);
24
25 #endif /* HEADER_H_ */
26
```

softwareTester.h

```
1 /*****
2  * AUTHOR      : Saul Moreno
3  * STUDENT ID  : 269491
4  * ASSIGNMENT# : 4
5  * CLASS      : CS1C
6  * SECTION    : MW 5:00pm
7  * DUE DATE   : 3/27/2020
8  *****/
9
10 #ifndef SOFTWARETESTER_H_
11 #define SOFTWARETESTER_H_
12
13 #include "employee.h"
14
15 class SoftwareTester: public Employee
16 {
17     public:
18         /***/
19         ****CONSTRUCTOR/DESTRUCTOR**
20         *****/
21         SoftwareTester(); //DEFAULT CONSTRUCTOR
22         ~SoftwareTester(); //DEFAULT DESTRUCTOR
23         void Address(SoftwareTester *tPtr, string add, string cit,
24                     string st, string zp);
25         void DeepCopy(SoftwareTester otherTester); // deep copy operation
26         void CopyCons(const SoftwareTester &otherTester); // copy constructor
27         void Display() const; // Will display employee information
28
29
30         /***/
31         **** MUTATORS ****
32         *****/
33     private:
34         char *addressPtr; // Pointer that points to the address of the employee
35         char *cityPtr;    // Pointer that points to the city of the employee
36         char *statePtr;   // Pointer that points to the state of the employee
37         char *zipPtr;     // Pointer that points to the zip code of the employee
38         string address;   // String that holds the
39         string city;      // String that holds the
40         string state;     // String that holds the
41         string zip;       // String that holds the
42         const int ADD_SIZE = 21; //Column size for the address
43         const int CITY_SIZE = 16; //Column size for the city
44         const int ST_SIZE = 6; //Column size for the state
45
46 };
47
48
49
50 #endif /* SOFTWARETESTER_H_ */
51
```

softwareTester.cpp

```
1 /*****
2  * AUTHOR      : Saul Moreno
3  * STUDENT ID  : 269491
4  * ASSIGNMENT# : 4
5  * CLASS       : CS1C
6  * SECTION    : MW 5:00pm
7  * DUE DATE    : 3/27/2020
8  *****/
9
10 #include "softwareTester.h"
11 #include <iomanip>
12 #include <iostream>
13 SoftwareTester::SoftwareTester()
14 {
15
16 }
17
18 SoftwareTester::~SoftwareTester()
19 {
20
21 }
22
23 void SoftwareTester::Address(SoftwareTester *tPtr, string add, string cit,
24                               string st, string zp)
25 {
26     tPtr->address = add;
27     tPtr->city = cit;
28     tPtr->state = st;
29     tPtr->zip = zp;
30     cout << std::left;
31     std::cout << std::setw(ADD_SIZE)<< tPtr->address;
32     std::cout << std::setw(CITY_SIZE)<< tPtr->city;
33     std::cout << std::setw(ST_SIZE)<< tPtr->state;
34     std::cout << tPtr->zip;
35     cout << std::right;
36 }
37
38 void SoftwareTester::DeepCopy(SoftwareTester otherTester)
39 {
40     address = otherTester.address;
41     city = otherTester.city;
42     state = otherTester.state;
43     zip = otherTester.zip;
44 }
45
46 void SoftwareTester::CopyCons(const SoftwareTester &otherTester)
47 {
48     address = otherTester.address;
49     city = otherTester.city;
50     state = otherTester.state;
51     zip = otherTester.zip;
52 }
53
54 void SoftwareTester::Display() const
55 {
56     cout << std::left;
57     std::cout << std::setw(ADD_SIZE)<< "Address";
```

softwareTester.cpp

```
58     std::cout << std::setw(CITY_SIZE)<< "City";
59     std::cout << std::setw(ST_SIZE)<< "State";
60     std::cout << "Zip Code";
61     cout << std::right;
62 }
63
```

employee.h

```
1 /*****
2  * AUTHOR      : Saul Moreno
3  * STUDENT ID  : 269491
4  * ASSIGNMENT# : 4
5  * CLASS       : CS1C
6  * SECTION     : MW 5:00pm
7  * DUE DATE    : 3/27/2020
8  *****/
9
10 #ifndef EMPLOYEE_H_
11 #define EMPLOYEE_H_
12
13 #include "header.h"
14
15 class Employee
16 {
17     public:
18         /***/
19         ****CONSTRUCTOR/DESTRUCTOR**
20         *****/
21         Employee(); //DEFAULT CONSTRUCTOR
22         ~Employee(); //DEFAULT DESTRUCTOR
23
24         /***/
25         **** MUTATORS ****
26         *****/
27         void SetInitial(string eName, int eId, string ePhone, int eAge,
28             char eGender, string eTitle, double eSalary, string eHire);
29         void ChangeEmployeesName(string eName); //changes the employee's name
30         void ChangeEmployeesId(int eId); //changes the employee's id
31         void ChangeEmployeesPhone(string ePhone); //changes the employee's phone #
32         void ChangeEmployeesAge(int eAge); //changes the employee's age
33         void ChangeEmployeesGender(char eGender); //changes the gender
34         void ChangeEmployeesJobTitle(string eTitle); //changes the employee's job
35             // title
36         void ChangeEmployeesSalary(double eSalary); //changes the employee's salary
37         void ChangeEmployeesHireDate(string eHire); //changes the employee's hire
38             // date
39
40         /***/
41         **** ACCESSORS ****
42         *****/
43         void DisplayTable() const; // Will display table
44         void Display() const; // Will display employee information
45
46     protected:
47         const int SET_NAME = 15; // - column width for the name
48         const int SET_ID = 14; // - column width for the id
49         const int SET_PHONE = 14; // - column width for the phone
50         const int SET_AGE = 5; // - column width for the age
51         const int SET_GEN = 7; // - column width for the gender
52         const int SET_TITLE = 16; // - column width for the job title
53         const int SET_SAL = 10; // - column width for the salary
54         const int SET_DATE = 8; // - column width for the date
55         string employeeName; // - holds employee's name
56
57     private:
```


employee.h

```
58  int    employeeId;    // - holds employee's id
59  string phoneNumber;   // - holds employee's phone number
60  int    employeeAge;   // - holds employee's age
61  char   employeeGender; // - holds employee's gender
62  string employeeTitle; // - holds employee's title
63  double employeeSalary; // - holds employee's salary
64  string hireDate;      // - holds employee's hire date
65
66 };
67
68 #endif /* EMPLOYEE_H_ */
69
```

employee.cpp

```
1 /*****
2  * AUTHOR      : Saul Moreno
3  * STUDENT ID  : 269491
4  * ASSIGNMENT# : 4
5  * CLASS       : CS1C
6  * SECTION     : MW 5:00pm
7  * DUE DATE    : 3/27/2020
8  *****/
9
10 #include "employee.h"
11
12 Employee::Employee()
13 {
14     employeeName = "Saul Moreno";
15     employeeId = 255; // - sets employee's id
16     phoneNumber = "523-545-5254";
17     employeeAge = 0; // - sets employee's age
18     employeeGender = 'M'; // - sets employee's gender
19     employeeTitle = " ";
20     employeeSalary = 0; // - holds employee's salary
21     hireDate = "12/12/21";
22 }
23
24 Employee::~Employee()
25 {
26
27 }
28
29 /*****
30 *FUNCTION - SetInitial
31 *
32 *This function set the initial values for the employee
33 *
34 *PRE-CONDITIONS
35 * eName : has to be previously defined.
36 * eId : has to be previously defined.
37 * ePhone : has to be previously defined.
38 * eAge : has to be previously defined.
39 * eGender: has to be previously defined.
40 * eTitle : has to be previously defined.
41 * eSalary: has to be previously defined.
42 * eHire : has to be previously defined.
43 *POST-CONDITIONS
44 * employeeName - sets the new value for employee's name.
45 * employeeId - sets the new value for employee's id.
46 * phoneNumber - sets the new value for employee's phone number.
47 * employeeAge - sets the new value for employee's age.
48 * employeeGender - sets the new value for employee's gender.
49 * employeeTitle - sets the new value for employee's title.
50 * employeeSalary - sets the new value for employee's salary
51 * hireDate - sets the new value for employee's hire date.
52 *****/
53 void Employee::SetInitial(string eName, int eId, string ePhone, int eAge,
54                             char eGender, string eTitle, double eSalary, string eHire)
55 {
56     employeeName = eName;
57     employeeId = eId;
```

employee.cpp

```
58     phoneNumber = ePhone;
59     employeeAge = eAge;
60     employeeGender = eGender;
61     employeeTitle = eTitle;
62     employeeSalary = eSalary;
63     hireDate = eHire;
64 }
65
66 /*****
67 *FUNCTION - ChangeEmployeesName
68 *
69 *This function will change the employee's name.
70 *
71 *PRE-CONDITIONS
72 * eName: has to be previously defined.
73 *
74 *POST-CONDITIONS
75 * employeeName - sets the new value for employee's name.
76 *
77 *****/
78 void Employee::ChangeEmployeesName(string eName)
79 {
80     employeeName = eName;
81 }
82
83 /*****
84 *FUNCTION - ChangeEmployeesId
85 *
86 *This function will change the employee's ID.
87 *
88 *PRE-CONDITIONS
89 * eId: has to be previously defined.
90 *
91 *POST-CONDITIONS
92 * employeeId - sets the new value for the employee's ID.
93 *
94 *****/
95 void Employee::ChangeEmployeesId(int eId)
96 {
97     employeeId = eId;
98 }
99
100 /*****
101 *FUNCTION - ChangeEmployeesPhone
102 *
103 *This function changes the employee's phone number.
104 *
105 *PRE-CONDITIONS
106 * ePhone: has to be previously defined.
107 *
108 *POST-CONDITIONS
109 * phoneNumber - sets the new value for the employee's phone number.
110 *
111 *****/
112 void Employee::ChangeEmployeesPhone(string ePhone)
113 {
114     phoneNumber = ePhone;
```

employee.cpp

```
115 }
116
117 /*****
118 *FUNCTION - ChangeEmployeesAge
119 *
120 *This function changes the employee's age.
121 *
122 *PRE-CONDITIONS
123 * eAge: has to be previously defined.
124 *
125 *POST-CONDITIONS
126 * employeeAge - sets the new value for the employee.
127 *
128 *****/
129 void Employee::ChangeEmployeesAge(int eAge)
130 {
131     employeeAge = eAge;
132 }
133
134 /*****
135 *FUNCTION - ChangeEmployeesGender
136 *
137 *This function will change the employee's gender.
138 *
139 *PRE-CONDITIONS
140 * eGender: has to be previously defined.
141 *
142 *POST-CONDITIONS
143 * employeeGender - sets the new value for employee's gender.
144 *
145 *****/
146 void Employee::ChangeEmployeesGender(char eGender)
147 {
148     employeeGender = eGender;
149 }
150
151 /*****
152 *FUNCTION - ChangeEmployeesJobTitle
153 *
154 *This function will set employee's job title.
155 *
156 *PRE-CONDITIONS
157 * eTitle: has to be previously defined.
158 *
159 *POST-CONDITIONS
160 * employeeTitle - stores the new value for employee's job title.
161 *
162 *****/
163 void Employee::ChangeEmployeesJobTitle(string eTitle)
164 {
165     employeeTitle = eTitle;
166 }
167
168 /*****
169 *FUNCTION - ChangeEmployeesSalary
170 *
171 *This function will change the employee's salary.
```

employee.cpp

```
172 *
173 *PRE-CONDITIONS
174 * eSalary: has to previously be defined.
175 *
176 *POST-CONDITIONS
177 * employeeSalary - stores the new value for employees salary.
178 *
179 *****/
180 void Employee::ChangeEmployeesSalary(double eSalary)
181 {
182     employeeSalary = eSalary;
183 }
184
185 /*****
186 *FUNCTION - ChangeEmployeesHireDate
187 *
188 *This function will change the employee's hire date.
189 *
190 *PRE-CONDITIONS
191 * eHire: has to previously be defined.
192 *
193 *POST-CONDITIONS
194 * hireDate - stores the new value for the hire date.
195 *
196 *****/
197 void Employee::ChangeEmployeesHireDate(string eHire)
198 {
199     hireDate = eHire;
200 }
201
202 /*****
203 *FUNCTION - DisplayTable
204 *
205 *This function will print out the table heading.
206 *
207 *PRE-CONDITIONS
208 * This is a void function
209 *
210 *POST-CONDITIONS
211 * This will not change anything member variables
212 *
213 *****/
214 void Employee::DisplayTable() const
215 {
216     #include <iomanip>
217     cout << left;
218
219     cout << setw(SET_NAME) << "Name" << setw(SET_ID) << "Employee's Id"
220         << setw(SET_PHONE) << "Phone" << setw(SET_AGE) << "Age"
221         << setw(SET_GEN) << "Gender" << setw(SET_TITLE) << "Job Title"
222         << setw(SET_SAL) << "Salary" << setw(SET_DATE) << "Date" << endl;
223     cout << right;
224 }
225
226 /*****
227 *FUNCTION - Display
228 *
```

employee.cpp

```
229 *This function will print out all the information that was stored.
230 *
231 *PRE-CONDITIONS
232 * This is a void function
233 *
234 *POST-CONDITIONS
235 * This will not change anything member variables
236 *
237 *****/
238 void Employee::Display() const
239 {
240     #include <iomanip>
241
242     cout << left;
243     cout << setprecision(7);
244
245     cout << setw(SET_NAME) << employeeName
246           << setw(SET_ID) << employeeId
247           << setw(SET_PHONE) << phoneNumber
248           << setw(SET_AGE) << employeeAge
249           << setw(SET_GEN) << employeeGender
250           << setw(SET_TITLE) << employeeTitle
251           << '$' << setw(SET_SAL) << employeeSalary
252           << setw(SET_DATE) << hireDate;
253     cout << right;
254     cout << setprecision(6);
255 }
256
```

programmer.h

```
1 /*****
2  * AUTHOR      : Saul Moreno
3  * STUDENT ID  : 269491
4  * ASSIGNMENT# : 4
5  * CLASS       : CS1C
6  * SECTION     : MW 5:00pm
7  * DUE DATE    : 3/27/2020
8  *****/
9
10 #ifndef PROGRAMMER_H_
11 #define PROGRAMMER_H_
12
13 #include "employee.h"
14
15 class Programmer: public Employee
16 {
17     public:
18         /***/
19         ****CONSTRUCTOR/DESTRUCTOR**
20         *****/
21         Programmer(); //DEFAULT CONSTRUCTOR
22         ~Programmer(); //DEFAULT DESTRUCTOR
23
24         /***/
25         **** MUTATORS ****
26         *****/
27         void SetInitial(string eName, int dNumber, string sName, int pInc,
28             string cIdn, string javId);
29         void ChangeDepartmentNum(int dNum); //changes the employee's department #
30         void ChangeSupervisorName(string sName); //changes the supervisors name
31         void ChangePercentInc(int pInc); //changes the employee's salary % inc
32         void ChangeCppIdentifier(string cIden); //status of C++ knowledge
33         void ChangeJavaIdentifier(string jIden); //status of Java knowledge
34
35         /***/
36         **** ACCESSORS ****
37         *****/
38         void DisplayTable() const; // Will display table
39         void Display() const; // Will display employee information
40
41     private:
42         int departNumber; // - sets employee's department name
43         string supervisorName; // - sets employee's supervisors name
44         int percentInc; // - sets employee's percent increase
45         string cppIden; // - sets employee's c++ knowledge
46         string javaIden; // - sets employee's java knowledge
47
48 };
49
50 #endif /* PROGRAMMER_H_ */
51
```

programmer.cpp

```
1 /*****
2  * AUTHOR      : Saul Moreno
3  * STUDENT ID  : 269491
4  * ASSIGNMENT# : 4
5  * CLASS       : CS1C
6  * SECTION     : MW 5:00pm
7  * DUE DATE    : 3/26/2020
8  *****/
9
10 #include "programmer.h"
11
12 Programmer::Programmer()
13 {
14     departNumber = 0; // - sets employee's department name
15     percentInc = 0;   // - sets employee's percent increase
16     cppIdn = 'Y';     // - sets employee's c++ knowledge
17     javaIdn = 'Y';
18 }
19
20 Programmer::~Programmer()
21 {
22
23 }
24
25 /*****
26 *FUNCTION - SetInitial
27 *
28 *This function set the initial values for the employee
29 *
30 *PRE-CONDITIONS
31 * dNumber: has to be previously defined.
32 * sName : has to be previously defined.
33 * pInc : has to be previously defined.
34 * cIdn : has to be previously defined.
35 * javId : has to be previously defined.
36 *POST-CONDITIONS
37 * departNumber - sets the new value for employee's department number.
38 * supervisorName - sets the new value for employee's supervisors name.
39 * percentInc - sets the new value for employee's salary % increase.
40 * cppIdn - sets the new value for employee's C++ knowledge status(Y/N).
41 * javaIdn - sets the new value for employee's Java knowledge status(Y/N).
42 *****/
43 void Programmer::SetInitial(string eName, int dNumber, string sName, int pInc,
44                             string cIdn, string javId)
45 {
46     Employee::ChangeEmployeesName(eName);
47     departNumber = dNumber;
48     supervisorName = sName;
49     percentInc = pInc;
50     cppIdn = cIdn;
51     javaIdn = javId;
52 }
53
54 /*****
55 *FUNCTION - CdepartmentNum
56 *
57 *This function changes the employee's department number.
```


programmer.cpp

```
58 *
59 *PRE-CONDITIONS
60 * dNum: has to be previously defined.
61 *
62 *POST-CONDITIONS
63 * departmentNumber- sets the new value for the employee's department number.
64 *
65 *****/
66 void Programmer::ChangeDepartmentNum(int dNum)
67 {
68     departNumber = dNum;
69 }
70
71 /*****
72 *FUNCTION - CsupervisorName
73 *
74 *This function changes the employee's supervisors name.
75 *
76 *PRE-CONDITIONS
77 * sName: has to be previously defined.
78 *
79 *POST-CONDITIONS
80 * supervisorName - sets the new value for the employee's supervisor.
81 *
82 *****/
83 void Programmer::ChangeSupervisorName(string sName)
84 {
85     supervisorName = sName;
86 }
87
88 /*****
89 *FUNCTION - CpercentInc
90 *
91 *This function changes the employee's salary percent increase.
92 *
93 *PRE-CONDITIONS
94 * pInc: has to be previously defined.
95 *
96 *POST-CONDITIONS
97 * percentInc- sets the new value for the employee's percent increase.
98 *
99 *****/
100 void Programmer::ChangePercentInc(int pInc)
101 {
102     percentInc = pInc;
103 }
104
105 /*****
106 *FUNCTION - CcppIdentifier
107 *
108 *This function changes the employee's c++ status to (Y/N).
109 *
110 *PRE-CONDITIONS
111 * cIdent: has to be previously defined.
112 *
113 *POST-CONDITIONS
114 * cppIden- sets a (Y/N) value.
```

programmer.cpp

```
115 *
116 *****/
117 void Programmer::ChangeCppIdentifier(string cIden)
118 {
119     cppIden = cIden;
120 }
121
122 /*****
123 *FUNCTION - ChangeEmployeesPhone
124 *
125 *This function changes the employee's java status (Y/N).
126 *
127 *PRE-CONDITIONS
128 * jIden: has to be previously defined.
129 *
130 *POST-CONDITIONS
131 * javaIden- sets a (Y/N) value.
132 *
133 *****/
134 void Programmer::ChangeJavaIdentifier(string jIden)
135 {
136     javaIden = jIden;
137 }
138
139 /*****
140 *FUNCTION - DisplayTable
141 *
142 *This function will print out the table heading.
143 *
144 *PRE-CONDITIONS
145 * This is a void function
146 *
147 *POST-CONDITIONS
148 * This will not change anything member variables
149 *
150 *****/
151 void Programmer::DisplayTable() const
152 {
153     #include <iomanip>
154     cout << left;
155
156     cout << setw(SET_NAME) << "Name" << setw(SET_ID) << "Department"
157         << setw(18) << "Supervisor Name" << setw(SET_DATE) << "Raise %"
158         << setw(SET_TITLE) << "C++ Knowledge" << setw(SET_ID)
159         << "Java Knowledge" << endl;
160     cout << right;
161 }
162
163 /*****
164 *FUNCTION - Display
165 *
166 *This function will print out all the information that was stored.
167 *
168 *PRE-CONDITIONS
169 * This is a void function
170 *
171 *POST-CONDITIONS
```

programmer.cpp

```
172 * This will not change anything member variables
173 *
174 *****/
175 void Programmer::Display() const
176 {
177     string eName;
178     //Employee::Display();
179     //Employee::ChangeEmployeesName(eName);
180 #include <iomanip>
181     cout << left;
182
183     cout << setw(SET_NAME) << employeeName << setw(SET_ID) << departNumber
184         << setw(18) << supervisorName << setw(SET_DATE) << percentInc
185         << setw(SET_TITLE) << cppIden << setw(SET_ID)
186         << javaIden << endl;
187     cout << right;
188
189 }
190
```

softwareArchitect.h

```
1 /*****
2  * AUTHOR      : Saul Moreno
3  * STUDENT ID  : 269491
4  * ASSIGNMENT# : 4
5  * CLASS       : CS1C
6  * SECTION     : MW 5:00pm
7  * DUE DATE    : 3/27/2020
8  *****/
9
10 #ifndef SOFTWAREARCHITECT_H_
11 #define SOFTWAREARCHITECT_H_
12
13 #include "employee.h"
14
15 class SoftwareArchitect: public Employee
16 {
17     public:
18         /***/
19         ****CONSTRUCTOR/DESTRUCTOR**
20         *****/
21         SoftwareArchitect(); //DEFAULT CONSTRUCTOR
22         ~SoftwareArchitect(); //DEFAULT DESTRUCTOR
23
24         /***/
25         **** MUTATORS ****
26         *****/
27         void SetInitial(string eName, int dNum, string sName, int sPer,
28             int nExp); //sets the employee's value
29         void ChangeDepartNum(int dNum); // changes department name
30         void ChangeSuperName(string sName); // changes supervisors name
31         void ChangeSalPer(int sPer); //changes salary percent raise
32         void ChangeNumExp(int nExp); //changes employees's years of experience
33         /***/
34         **** ACCESSORS ****
35         *****/
36         void DisplayTable() const; // Will display table heading
37         void Display() const; // Will display employee information
38
39     private:
40         int departmentNumber; // - sets employee's department name
41         string supervisorName; // - sets employee's supervisors name
42         int salaryPercentInc; // - sets the salary % increase
43         int numExp; // - sets the number of experience
44 };
45
46 #endif /* SOFTWAREARCHITECT_H_ */
47
```

softwareArchitect.cpp

```
1 /*****
2  * AUTHOR      : Saul Moreno
3  * STUDENT ID  : 269491
4  * ASSIGNMENT# : 4
5  * CLASS       : CS1C
6  * SECTION     : MW 5:00pm
7  * DUE DATE    : 3/27/2020
8  *****/
9
10 #include "softwareArchitect.h"
11
12 SoftwareArchitect::SoftwareArchitect()
13 {
14     departmentNumber = 0;
15     salaryPercentInc = 0;
16     numExp           = 0;
17 }
18
19 SoftwareArchitect::~SoftwareArchitect()
20 {
21
22 }
23
24 /*****
25 *FUNCTION - SetInitial
26 *
27 *This function set the initial values for the employee
28 *
29 *PRE-CONDITIONS
30 * eName : has to be previously defined.
31 * dNum  : has to be previously defined.
32 * sName : has to be previously defined.
33 * sPer  : has to be previously defined.
34 * nExp  : has to be previously defined.
35 *POST-CONDITIONS
36 * employeeName - sets the new value for employee's name.
37 * departNumber - sets the new value for employee's department number.
38 * supervisorName - sets the new value for employee's supervisors name.
39 * salaryPercentInc - sets the new value for employee's salary % increase.
40 * numExp - sets the new value for employee's # of experience.
41 *****/
42 void SoftwareArchitect::SetInitial(string eName, int dNum, string sName,
43                                     int sPer, int nExp)
44 {
45     Employee::ChangeEmployeesName(eName);
46     departmentNumber = dNum;
47     supervisorName = sName;
48     salaryPercentInc = sPer;
49     numExp = nExp;
50 }
51
52 /*****
53 *FUNCTION - ChangeDepartNum
54 *
55 *This function changes the employee's department number.
56 *
57 *PRE-CONDITIONS
```

softwareArchitect.cpp

```
58 * dNum: has to be previously defined.
59 *
60 *POST-CONDITIONS
61 * departNumber- sets the new value for employee's department number.
62 *
63 *****/
64 void SoftwareArchitect::ChangeDepartNum(int dNum)
65 {
66     departmentNumber = dNum;
67 }
68
69 /*****
70 *FUNCTION - ChangeSuperName
71 *
72 *This function change the employee's supervisors name.
73 *
74 *PRE-CONDITIONS
75 * sName: has to be previously defined.
76 *
77 *POST-CONDITIONS
78 * supervisorName- sets the new value for employee's supervisors name.
79 *
80 *****/
81 void SoftwareArchitect::ChangeSuperName(string sName)
82 {
83     supervisorName = sName;
84 }
85
86 /*****
87 *FUNCTION - ChangeSalPer
88 *
89 *This function will change the employee's salary percent increase.
90 *
91 *PRE-CONDITIONS
92 *sName: has to be previously defined.
93 *
94 *POST-CONDITIONS
95 *supervisorName- sets the new value for employee's supervisors name.
96 *
97 *****/
98 void SoftwareArchitect::ChangeSalPer(int sPer)
99 {
100     salaryPercentInc = sPer;
101 }
102
103 /*****
104 *FUNCTION - ChangeNumExp
105 *
106 *This function will change the number of years of experience that the
107 *employee has.
108 *
109 *PRE-CONDITIONS
110 * nExp: has to be previously defined.
111 *
112 *POST-CONDITIONS
113 * numExp - sets the new value for employee's # of experience.
114 *
```

softwareArchitect.cpp

```
115 *****/
116 void SoftwareArchitect::ChangeNumExp(int nExp)
117 {
118     numExp = nExp;
119 }
120
121 *****/
122 *FUNCTION - DisplayTable
123 *
124 *This function will print out the table heading.
125 *
126 *PRE-CONDITIONS
127 * This is a void function
128 *
129 *POST-CONDITIONS
130 * This will not change anything member variables
131 *
132 *****/
133 void SoftwareArchitect::DisplayTable() const
134 {
135     #include <iomanip>
136     cout << left;
137
138     cout << setw(SET_NAME) << "Name" << setw(SET_ID) << "Department"
139         << setw(17) << "Supervisor Name" << setw(6) << "Raise"
140         << setw(SET_GEN) << "Years of Experience" << endl;
141     cout << right;
142 }
143
144 *****/
145 *FUNCTION - Display
146 *
147 *This function will print out all the information that was stored.
148 *
149 *PRE-CONDITIONS
150 * This is a void function
151 *
152 *POST-CONDITIONS
153 * This will not change anything member variables
154 *
155 *****/
156 void SoftwareArchitect::Display() const
157 {
158     #include <iomanip>
159     cout << left;
160
161     cout << setw(SET_NAME) << employeeName << setw(SET_ID) << departmentNumber
162         << setw(17) << supervisorName << setw(6) << salaryPercentInc
163         << setw(SET_GEN) << numExp << endl;
164     cout << right;
165 }
166
```

date.h

```
1 /*****
2  * AUTHOR      : Saul Moreno
3  * STUDENT ID  : 269491
4  * ASSIGNMENT# : 4
5  * CLASS       : CS1C
6  * SECTION     : MW 5:00pm
7  * DUE DATE    : 3/27/2020
8  *****/
9
10 #ifndef DATE_H_
11 #define DATE_H_
12
13 #include "header.h"
14
15 class Date
16 {
17     public:
18     Date();
19     ~Date();
20
21     void ChangeDate(int nMonth, int nDay, int nYear); //Changes the month, day
22                                                    // and year
23
24     private:
25     int month; // holds the value for the month
26     int day;   // holds the value for the day
27     int year;  // holds the value for the year
28 };
29
30 #endif /* DATE_H_ */
31
```


date.cpp

```
1 /*****
2  * AUTHOR      : Saul Moreno
3  * STUDENT ID  : 269491
4  * ASSIGNMENT# : 4
5  * CLASS       : CS1C
6  * SECTION     : MW 5:00pm
7  * DUE DATE    : 3/27/2020
8  *****/
9
10 #include "date.h"
11
12 Date::Date()
13 {
14     month = 0;
15     day   = 0;
16     year  = 0;
17 }
18
19 Date::~Date()
20 {
21
22 }
23
24 /*****
25 *FUNCTION - ChangeDate
26 *
27 *This function will change the employee's hire date.
28 *
29 *PRE-CONDITIONS
30 * nMonth: has to be previously defined.
31 * nDay   : has to be previously defined.
32 * nYear  : has to be previously defined.
33 *
34 *POST-CONDITIONS
35 * nMonth - sets the new value for employee's month.
36 * nDay   - sets the new value for employee's day.
37 * nYear  - sets the new value for employee's year.
38 *
39 *****/
40 void Date::ChangeDate(int nMonth, int nDay, int nYear)
41 {
42     month = nMonth;
43     day   = nDay;
44     year  = nYear;
45
46 }
47
```

PrintHeader.cpp

```
1 /*****
2  * AUTHOR      : Saul Moreno
3  * STUDENT ID  : 269491
4  * ASSIGNMENT# : 4
5  * CLASS       : CS1C
6  * SECTION     : MW 5:00pm
7  * DUE DATE    : 3/27/2020
8  *****/
9
10 #include "header.h"
11
12 /*****
13  *FUNCTION - PrintHeader
14  *
15  *This function receives an assignment name, type and number then outputs the
16  * appropriate header - returns nothing.
17  *
18  *PRE-CONDITIONS
19  *   asName: Has to be previously defined
20  *   asType: Has to be previously defined
21  *   asNum: Has to be previously defined
22  *
23  *POST-CONDITIONS
24  *   This function will output class heading.
25  *   <Post-conditions are the changed outputs either passed by value or
26  *   by reference OR anything affected by the function.
27  *
28  *****/
29
30 void PrintHeader(string asName, // IN - Assignment Name
31                  int asNum,     // IN - assignment type
32                  // (LAB or ASSIGNMENT)
33                  char asType) // IN - assignment number
34 {
35     cout << left;
36     cout << "*****\n";
37     cout << "* PROGRAMMED BY : Saul Moreno\n";
38     cout << "* "<< setw(14) << "STUDENT ID" << ": 269491\n";
39     cout << "* "<< setw(14) << "CLASS" << ": CS 1B MW-7:30pm\n";
40     cout << "* ";
41     if (toupper(asType) == 'L')
42     {
43         cout << "LAB #" << setw(9);
44     }
45     else
46     {
47         cout << "ASSIGNMENT #" << setw(2);
48     }
49     cout << asNum << ": " << asName << endl;
50     cout << "*****\n\n";
51     cout << right;
52 }
53
```