```cpp
/*****************************************************************************
 * PROGRAMMER : Ali Eshghi
 * STUDENT ID : 1112261
 * CLASS      : CS1C
 * SECTION    : MW 5pm
 * Assign #4  : Enhanced Employee
 * DUE DATE   : 26 Febuary 2020
 *****************************************************************************/

#include"MyHeader.h"

/*****************************************************************************
 * Methods for class date
 *****************************************************************************/

//non-Default constructor
date::date()
{
    //INITIALIZATION
    day   = 0;
    month = 0;
    year  = 0;
}

//destruactor
date::~date() {}


/*****************************************************************************
 * Methods for class employee
 *****************************************************************************/

//default constructor
employee::employee()
{
    //INITIALIZATION
    name.clear();
    id.clear();
    phoneNum.clear();
    jobTitle.clear();

    age    = 0;
    salary = 0;

    gender = ' ';

}

//destructor
employee::~employee() {}

//method for getting the name from the client and store it in name attribute
void employee::setName(string empName)
{
    name = empName;
}

string employee::getName()
{
```

```cpp
    return name;
}

//methpod for getting the id from the client and store it in id attribute
void employee::setId(string empId)
{
    id = empId;
}

//method for getting the phone number from the client and store it in
//the phoenNum attribute
void employee::setPhoneNumber(string number)
{
    phoneNum = number;
}

//method for getting the age from the client and store it in
//age attribute
void employee::setAge(int empAge)
{
    age = empAge;
}
//method for getting the gender from the client and store it
//in gender attribute
void employee::setGender(char sex)
{
    gender = sex;
}

//method for getting the job title from the client and store
//it in the jobTile attribute
void employee::setJobTitle(string title)
{
    jobTitle = title;
}

//method for getting the salary from the client and store it
//in salary attribute
void employee::setSalary(double income)
{
    salary = income;
}

//method for getting the hire date attributes and save the date
//into the attributes of day, month, and year
void employee::setDate(int startDay, int startMonth, int startYear)
{
    day   = startDay;
    month = startMonth;
    year  = startYear;
}

//method for printing the attributes with the informations stored
//in them from the client to the screen
void employee::print()
{

    cout << left;
    cout << fixed <<setprecision(2);
```

```
119     cout << setw(15)  << name      << setw(9) << id      << setw(16)
120          << phoneNum  << setw(8)   << age      << setw(7) << gender
121          << setw(15)  << jobTitle  << "$" << setw(15) << salary << month << "/"
122          << day << "/" << year << endl;
123 }
124
125
126
127 /*****************************************************************************
128  * Methods for class softTester
129  *****************************************************************************/
130
131 //Default cosntructor
132 softTester::softTester(string defAddress, string defCity, string defState, int
    defZipCode)
133 {
134     cout << "\nNormal constructor allocating ptr." << endl;
135
136     //allocate memory for the pointer
137
138     address = new string;
139     *address = defAddress;
140
141     city = new string;
142     *city = defCity;
143
144     state = new string;
145     *state = defState;
146
147     zipCode = new int;
148     *zipCode = defZipCode;
149 }
150
151
152 //copy constructor
153 softTester::softTester(const softTester& obj)
154 {
155     //using the deep copying to copy
156     cout << "\nCopy constructor allocating the ptr. " << endl;
157
158     //first we need to deallocate any value that this string is holding
159     delete address;
160     delete city;
161     delete state;
162     delete zipCode;
163
164     //address is a pointer, so we need to deep copy it if it is non-null
165     if(obj.address)
166     {
167         //allocate memory for our copy
168         address = new string;
169         city    = new string;
170         state   = new string;
171         zipCode = new int;
172
173         //do the copy
174         address = obj.address;
175         city    = obj.city;
176         state   = obj.state;
```

```
177         zipCode = obj.zipCode;
178     }
179
180     else
181     {
182         address = NULL;
183         city    = NULL;
184         state   = NULL;
185         zipCode = NULL;
186     }
187 }
188
189 //destructor
190 softTester::~softTester(void)
191 {
192     cout << "\n\nfreeing memory" << endl << endl;
193     delete address;
194     delete city;
195     delete state;
196     delete zipCode;
197 }
198
199 //Method for getting the address,
200 //return type: String
201 string softTester::getAddress(void)
202 {
203     return *address;
204 }
205
206 //Method for getting the City,
207 //return type: String
208 string softTester::getCity(void)
209 {
210     return *city;
211 }
212
213 //Method for getting the State,
214 //return type: String
215 string softTester::getState(void)
216 {
217     return *state;
218 }
219
220 //Method for getting the zip code,
221 //return type: integer
222 int softTester::getZipCode(void)
223 {
224     return *zipCode;
225 }
226
227 //Method for changing the address attribute
228 //of the class softTester
229 void softTester::changeAddress(string newAddress)
230 {
231     address = new string;
232     *address = newAddress;
233
234 }
235
```

```cpp
//Method for changing the city attribute
//of the class softTester
void softTester::changeCity(string newCity)
{
    city = new string;
    *city = newCity;
}

//Method for changing the state attribute
//of the class softTester
void softTester::changeState(string newState)
{
    state = new string;
    *state = newState;
}

//Method for changing the zipCode attribute
//of the class softTester
void softTester::changeZipCode(int newZipCode)
{
    zipCode = new int;
    *zipCode = newZipCode;
}

//Method for diplayong the attributes
void softTester::softTesterDisplay()
{
    cout << left;
    cout << fixed <<setprecision(2);
    cout << setw(15)   << employee::getName()
         << setw(23)   << *address  << setw(16)
         << *city      << setw(8)   << *state    << setw(7) << *zipCode
         << endl;

}
```