```cpp
//Grant dawson COSC 220 Project 1
  #include "StudentDB.h"
//deconstructor
  StudentDB::~StudentDB(){
    studentNode* cursor=head;
    while(head){
      head=head->next;
      delete cursor;
      cursor=head;
    }
    delete [] major;
    delete [] grade;
    delete [] department;
  };

  //copy contructor
  StudentDB::StudentDB(const StudentDB& old){
    if(old.head==nullptr){
      head=nullptr;
    }else{
      studentNode* oldCursor=old.head;
      studentNode* cursor=head;
      while(oldCursor){
        addStudent(oldCursor->s);
        cursor->c=oldCursor->c;
        cursor=cursor->next;
        oldCursor=oldCursor->next;
      }
    }
  };

  //equal operator
  StudentDB* StudentDB::operator=(const StudentDB& x){
    studentNode* oldCursor=x.head;
    while(oldCursor){
      addStudent(oldCursor->s);
      oldCursor=oldCursor->next;
    }
      return (this);
  }

  StudentDB::StudentDB(){
    head=nullptr;
  }

//constructor with a pre given student
  StudentDB::StudentDB(Student t){
    studentNode* newNode=new studentNode;
    newNode->next=nullptr;
    newNode->s=t;
    head=newNode;
  }

//This will update a desired student with a new given student
  void StudentDB::updateStudent(Student replace, Student t){
    studentNode* cursor=head;
    if(!head){
      cout<<"No students in the list"<<endl;
      return;
    }
    while(cursor){
      if(cursor->s==replace){
```

```
63          cursor->s.setName(t.getName());
64          cursor->s.setDOB(t.getDOB());
65          cursor->s.setMajor(t.getMajor());
66          return;
67        }
68        cursor=cursor->next;
69      }
70      cout<<"No student updated!"<<endl;
71    }
72  //This will remove a give student
73    void StudentDB::removeStudent(Student rmS){
74      studentNode* cursor=head;
75      studentNode* prev=head;
76      if(!head){
77        cout<<"No students in the list"<<endl;
78        return;
79      }
80      if(head->s == rmS){
81        head=head->next;
82        delete cursor;
83        return;
84      }
85      while(cursor){
86        if(cursor->s==rmS){
87          prev->next=cursor->next;
88          delete cursor;
89          return;
90        }
91        prev=cursor;
92        cursor=cursor->next;
93      }
94      cout<<rmS.getName()<<" was not found it could not be deleted!"<<endl;
95    }
96
97  //This will remove a given student info from the list
98    void StudentDB::removeStudent(string n, string dob, string m){
99      Student temp(n,dob,m);
100     removeStudent(temp);
101   }
102
103   void StudentDB::addStudent(Student crS){
104     studentNode* cursor=head;
105     studentNode* newNode = new studentNode;
106     newNode->s=crS;
107     newNode->next=nullptr;
108     if(head==nullptr){
109       head=newNode;
110       return;
111     }
112     while(cursor->next){
113       cursor=cursor->next;
114     }
115     cursor->next=newNode;
116   }
117
118 //This will add a new student to the list
119   void StudentDB::addStudent(string n,string dob,string m){
120     Student temp(n,dob,m);
121     addStudent(temp);
122   }
123
124 //This will print only a specific student and it's courses
```

```
125   void StudentDB::printStudent(Student s){
126     studentNode* cursor=head;
127     if(!head){
128       cout<<"No students in the list"<<endl;
129       return;
130     }
131     while(cursor){
132       if(s == cursor->s){
133         cursor->s.print();
134         cursor->c.printAll();
135         return;
136       }
137       cursor=cursor->next;
138     }
139     cout<<"No student found with "<<s.getName()<<"'s info"<<endl;
140   }
141
142 //This prints all the students with thier courses
143   void StudentDB::printAllList(){
144     studentNode* cursor=head;
145     if(!head){
146       cout<<"No students in the list"<<endl;
147       return;
148     }
149     while(cursor){
150         cursor->s.print();
151         cursor->c.printAll();
152         cursor=cursor->next;
153     }
154   }
155
156   //This prings all the student names
157   void StudentDB::printAllStudents(){
158     studentNode* cursor=head;
159     if(!head){
160       cout<<"No students in the list"<<endl;
161     }
162     while(cursor){
163         cursor->s.print();
164         cursor=cursor->next;
165     }
166   }
167
168
169
170 //This will update a specific course to a specfic student
171   void StudentDB::updateCourse(Student t,Course z){
172     studentNode* cursor=head;
173     if(!head){
174       cout<<"No students in the list"<<endl;
175       return;
176     }
177     while(cursor){
178       if(t==cursor->s){
179         cursor->c.update(z,createCourse());
180         return;
181       }
182       cursor=cursor->next;
183     }
184   }
185
186   //This will remove a desired course form a specific student
```

```
187   void StudentDB::removeCourse(Student t, Course z){
188     studentNode* cursor=head;
189     if(!head){
190       cout<<"No students in the list"<<endl;
191       return;
192     }
193     while(cursor){
194       if(t==cursor->s){
195         cursor->c.remove(z);
196         return;
197       }
198       cursor=cursor->next;
199     }
200   }
201
202   //This will add a new course to a specific student give in the parameters
203   void StudentDB::addCourse(Student t, Course z){
204     studentNode* cursor=head;
205     if(!head){
206       cout<<"No students in the list"<<endl;
207       return;
208     }
209     while(cursor){
210       if(t==cursor->s){
211         cursor->c.append(z);
212         return;
213       }
214       cursor=cursor->next;
215     }
216   }
217
218 //This gives the length/amount of students in the linklist currently
219   int StudentDB::length(){
220     studentNode* cursor=head;
221     int counter=0;
222     if(!head)
223       return counter;
224     while(cursor){
225       counter++;
226       cursor=cursor->next;
227     }
228     return counter;
229   }
230
231
232
233
234
235 //This adds a ne wmajor to the major.txt file
236   void StudentDB::addMajorToList(string toAdd){
237       Mlength++;
238       int i=0;
239       ofstream input ("Majors.txt", std::ios_base::app);
240       input<<"\n"<<toAdd;
241       input.close();
242       readMajorList();
243     }
244
245     //This reads in the major choices form a file
246   void StudentDB::readMajorList(){
247       ifstream input ("Majors.txt");
248       int i=0;
```

```
249       string trash;
250       while(!input.eof()){
251         i++;
252         input>>trash;
253       }
254       input.close();
255       delete [] major;
256       ifstream reInput ("Majors.txt");
257       Mlength=i;
258       major = new string[i];
259       i--;
260       while(!reInput.eof()){
261         reInput>>major[i];
262         i--;
263       }
264       reInput.close();
265     }
266
267 //This returns a newly created student made form user input
268   Student StudentDB::createStudent(){
269     readMajorList();
270     cin.ignore(256, '\n');//NEEDED
271     cout<<"What is your name: "<<endl;
272     string n;
273     getline(cin,n);
274     cout<<"What is your date of birth: "<<endl;
275     string d;
276     getline(cin,d);
277     return createStudentMajor(n,d);
278   }
279
280   //This asks for what the user whishes to pick their major and if it is not listed will
      add it to the file
281   Student StudentDB::createStudentMajor(string n, string dob){
282     Student temp;
283     temp.setName(n);
284     temp.setDOB(dob);
285     for(int i=0; i<Mlength; i++){
286       cout<<i+1<<")"<<major[i]<<" ";
287     }
288     cout<<endl;
289     while(true){
290       cout<<"What is you major? If it is not listed type \"0\""<<endl;
291       int choice;
292       cin>>choice;
293       if(choice==0){
294         string newMajor;
295         cout<<"What is your major: "<<endl;
296         cin>>newMajor;
297         addMajorToList(newMajor);
298         temp.setMajor(newMajor);
299         return temp;
300       }else if(choice>0&&choice<=Mlength){
301         temp.setMajor(major[--choice]);
302         return temp;
303       }
304       cout<<choice<<" is not a valid choice!"<<endl;
305     }
306   }
307
308 //this functions returns a course created by the user.
309   Course StudentDB::createCourse(){
```

```
310     cin.ignore(256, '\n');//NEEDED
311     string dep=createCourseDepartment();
312     cout<<"what is the course number: "<<endl;
313     string num;
314     cin>>num;
315
316     cout<<"What semester did you take this: "<<endl;
317     cin.ignore(256, '\n');//NEEDED
318     string sem;
319     getline(cin,sem);
320     char grade=createCourseGrades();
321     Course temp(num,dep,sem,grade);
322     return temp;
323   }
324
325
326   //This akses the user what department they want to pick form the deparement array
327   string StudentDB::createCourseDepartment(){
328     //cin.ignore(256, 'n');//NEEDED
329     readDepartmentList();
330     for(int i=0; i<Dlength; i++){
331       cout<<i+1<<")"<<department[i]<<" ";
332     }
333     cout<<endl;
334     while(true){
335       cout<<"What department is you class in? If it is not listed type \"0\""<<endl;
336       int choice;
337       cin>>choice;
338       if(choice==0){
339         string newDep;
340         cout<<"What department is you class from: "<<endl;
341         cin>>newDep;
342         addMajorToList(newDep);
343         return newDep;
344       }else if(choice>0&&choice<=Dlength){
345         return department[--choice];
346       }
347       cout<<choice<<" is not a valid choice!"<<endl;
348     }
349   }
350
351   //if the user doesn't have a desired department in our database yet they can add it here
352   void StudentDB::addDepartmentToList(string toAdd){
353       Dlength++;
354       int i=0;
355       ofstream input ("Department.txt", std::ios_base::app);
356       input<<"\n"<<toAdd;
357       input.close();
358       readDepartmentList();
359     }
360
361     //This reads the deprtments form a file and then populates the department dynamic
      arrray
362   void StudentDB::readDepartmentList(){
363       ifstream input ("Department.txt");
364       int i=0;
365       string trash;
366       while(!input.eof()){
367         i++;
368         input>>trash;
369       }
370       input.close();
```

```
371       delete [] department;
372       ifstream reInput ("Department.txt");
373       Dlength=i;
374       department = new string[i];
375       i--;
376       while(!reInput.eof()){
377         reInput>>department[i];
378         i--;
379       }
380       reInput.close();
381     }
382
383     //This lists all the [oosible grades to pick from and returns what the user chose]
384   char StudentDB::createCourseGrades(){
385     //cin.ignore(256, 'n');//NEEDED
386     readGradeList();
387     for(int i=0; i<Glength; i++){
388         cout<<i+1<<")"<<grade[i]<<" ";
389     }
390     cout<<endl;
391     while(true){
392       cout<<"What was your grade: "<<endl;
393       int choice;
394       cin>>choice;
395       if(choice>0&&choice<=Glength){
396         return grade[--choice];
397       }
398       cout<<choice<<" is not a valid choice!"<<endl;
399     }
400   }
401
402   //reads in possible grades you got from a file and populates the grades dynamic array
403   void StudentDB::readGradeList(){
404     //cin.ignore(256, 'n');//NEEDED
405       ifstream input ("Grades.txt");
406       int i=0;
407       string trash;
408       while(!input.eof()){
409         i++;
410         input>>trash;
411       }
412       input.close();
413       delete [] grade;
414       ifstream reInput ("Grades.txt");
415       Glength=i;
416       grade = new char[i];
417       i--;
418       while(!reInput.eof()){
419         reInput>>grade[i];
420         i--;
421       }
422       reInput.close();
423     }
424
425     //This will return a student that the user picked out of the list of all possible
      students to pick
426     Student StudentDB::chooseStudent(){
427       cin.ignore(256, '\n');//NEEDED
428       studentNode* cursor=head;
429       Student fail("Empty","List","Detected");
430       if(!head){
431         //cout<<"No students in the list"<<endl;
```

```
432         return fail;
433       }
434       int i=0;
435       int choice;
436       while(cursor){
437           cout<<i+1<<")";
438           cursor->s.print();
439           cursor=cursor->next;
440           i++;
441       }
442       cout<<endl;
443       while(true){
444         cin>>choice;
445         if(choice>0&&choice<=i){
446           cursor=head;
447           for(int i=0;i<choice-1;i++){
448             cursor=cursor->next;
449           }
450           return cursor->s;
451         }
452         cout<<"Not a valid repsonse"<<endl;
453       }
454     }
455
456     //This returns a course that the user picked out of all the possibilities form the
        given student's node
457     Course StudentDB::chooseCourse(Student s){
458       cin.ignore(256, '\n');//NEEDED
459       studentNode* cursor=head;//TODO
460       Course fail("Empty","List","Detected",'F');
461       int index;
462       while(cursor){
463         if(cursor->s==s){
464           cursor->c.printAllList();
465           int numCourse=cursor->c.length();
466           int choice;
467           if(numCourse!=0){
468             while(true){
469               cin>>choice;
470               if(choice>0&&choice<=numCourse){
471                 return cursor->c.chooseCourse(--choice);
472               }
473               cout<<"Not a valid repsonse"<<endl;
474             }
475           }else{
476             return fail;
477           }
478         }else{
479           cursor=cursor->next;
480         }
481       }
482     }
483
```