Marisa Kuyava

CS 300

4-3 Milestone

Hash Table Pseudocode **Milestone 4-3.**

**//Create Class for Course**

class Course.{

string **variable** courseNumber

string **variable** courseName

vector of prerequisites

}

**//Create class for Hashtable**

Set unsigned int DEFAULT\_SIZE

Class Hashtable{

Private:

// Define structures to hold courses

Struct Node{

Course\* course

Unsigned int key

Node\* next

Node default constructor{

Key set equal to UINT\_MAX;

next set equal to nullpointer

}

//Initialize with a course

Node(Course courseNumber) : Node(){

course set equal to courseNumber

}

//Initialize with a course and key

Node(Course courseNumber, unsigned int aKey) : Node(course){

course set equal to courseNumber

key set equal to akey

}

}

Vector <Courses> courses;

Set unsigned int table size to DEFAULT\_SIZE

unsigned int hash(int key)

Public:

Hashtable default constructor

Int hashFunction (string key)

Void insertCourse(sting key)

}

**//Hash function**

Hash(Course courseNumber){

Return (courseNumber % DEFAULT\_SIZE)

}

**//Used to validate data for formatting errors before course is inserted**

**lineParser(vector<string> line){**

**if line.size() is equal to 2 line can be added as it has required format{**

**Create new course**

**Set courseNumber equal to line 0**

**Set courseName equal to line 1**

**Return new course**

**}**

**Else if line size is greaten than 2{**

**Create new course**

**Set courseNumber equal to line 0**

**Set courseName equal to line 1**

**for each additional line until the end of the vector{**

**pushback each line greater than 1 to prerequisite vector**

**}**

**Return new course**

**}**

**Else if line size is less than 2{**

**PRINT There is an error in the file format. Every course must have a course number and course name**

**}**

**}**

**//Insert Course into HashTable**

Insert(Course\* courseNumber){

Using hash function create key from courseNumber

Create keyNode to retrieve node via key created

If keynode is empty/null

Add course at current empty node

Else if keyNode is not empty

While loop through keyNodes linked list until an empty node is found

Add course at empty node found

}

//file loading

loadFile(file FileName){

Create hashtable

Create vector of strings to hold file data

String variable to hold each line

Open file with Ifstream

while get line finds a next line in the file{

stringstream stst (line)

while stst.good() is to true{

create variable to store substring of line

Use get line to break substring from string using comma delimitator

Push substring to temporary <string> vector

}

Insert temporary line vector to hashable using Insert Function and lineParser function

Clear temporary vector

}

}

Search(vector<Course> courses, string courseNumber){

searchKey : Use Hash function to generate key for courseNumber

**Create new course to hold course returned**

**For each Course in courses vector{**

**If current course key is equal to** searchKey{

Return current Course

}

}

Return empty

}

**//Print course number, name and prerequisites**

**Print (vector<Course> courses, string courseNumer){**

**Create new course to hold course returned**

**If course returned by search is empty{**

**Print ‘Course is not in the catalog’**

**Return**

**}**

**Else{**

**Print course’s number and Name**

**For each prerequisite in courses’s prerequisite vector{**

**Print prerequisite**

**}**

**}**

**}**

}