﻿**Project overview:** The project was to develop a course management system using python. This is a console-based application. There are eight functions to accomplish several actions, these are:

1. User Interface
2. Check Prerequisite
3. Add Course
4. Update Course
5. Delete Course
6. Show All Course
7. Search Course
8. Store Course to Text

The user interface displays a menu to the user that they can chose any function from given list or type “quit” in order to exit the system. Check prerequisite checks if the prerequisite is already in the course catalogue or not while adding prerequisite in add course. Add course prompt user to add a new course by taking course information like course code, course title, course credit hour/s and prerequisite/s from the user. Update course update the information of existing course. Delete course can be used to delete any existing course from the system. The show all course function displays the information of all the course in the system. Search course function can be used to search any specific course using the course code. Store course to text function store the course stored in the system to a text file so the user can have a physical copy in their hard drive.

**Project solution design:** In order to complete this project a class “Course” is created so that object of each course can be created. A dictionary information is taken to store course information such as course code, course title, course credit and course prerequisites. A function called user interface is created so that it can displays list of options available for user to select to perform certain task as well as to take input which option they want to choose or the user can type “quit” in order to exit the system. Check prerequisite check if the prerequisite is in the course catalogue or not it uses a for loop to search in the course catalogue if the course prerequisite is in the course code of any course. The add course function prompt user to give information such course code, course title, course credit hour/s and course prerequisite/s. If the prerequisite given is either “N/A” or something that already exist in the course catalogue then an object of Course class is created and it is stored in course catalogue list and the user is notified with a confirmation text but if the course prerequisite doesn’t exist add course is prompted and user is requested to add the course first before specifying it as prerequisite of any other course, this is done using the check prerequisite function inside the add course function.

For update course function the user first search the course he wants to update with the course code. A for loop is used in order to search the course in course catalogue list, a Boolean flag is used in order to identify if the course is found or not, if the flag returns true the course information is displayed and previous information such as course title, course credit and course prerequisites/s can be updated though prompted user inputs. If the flag shows false the system displays an error message for the user.

For Delete course function the user first search the course he wants to delete with the course code. A for loop is used in order to search the course in course catalogue list, a Boolean flag is used in order to identify if the course is found or not, if the flag returns true the course information is displayed. An input is taken to verify if the user really wants to deleted the course, if ‘y’ is given as input the course is deleted from the list using built-in remove function. If the course is not found the system displays an error message.

In the show all course function if the length of course catalogue list is zero it displays message to add course first, if the list is not empty with the help of for loop and range function it displays all the stored course in course catalogue.

Search course function take input from user as the course code and search the course catalogue list with a for loop and Boolean flag, if the searched course code matches any course in the course catalogue the flag returns true and information of that individual course is displayed if the course code doesn’t match it gives an error message and prompt to add course function.

The store course to text function uses the python file system to store the courses from course catalogue so the user can have a physical copy. First a text file is taken and open in writing mode, then using a for loop the information of course catalogue from top to bottom are written in the file. A conformation message is displayed once all the lines are written in the text file.

Finally using a while loop and if/elif condition user’s choice are taken from option one to six, the while loop is terminated once the input is “quit”.

**Implementation:**

class Course:

"""For storing information dictionary as object"""

def \_\_init\_\_(self,course\_code,course\_title,course\_credit,course\_prerequisite):

self.information = {}

self.information['course\_code'] = course\_code

self.information['course\_title'] = course\_title

self.information['course\_credit'] = course\_credit

self.information['course\_prerequisite'] = course\_prerequisite

user\_interface\_input = 0

def user\_interface ():

"""Displaying Menu/User Interface"""

print ("\nCOURSE MANAGEMENT SYSTEM\n")

print ("1. Add Course")

print ("2. Update Course")

print ("3. Delete Course")

print ("4. Show All Course")

print ("5. Search Course")

print ("6. Store Course Information To Text File\n")

user\_interface\_input =input("Chose An Option From 1-6 Or Press quit To Exit : ")

return user\_interface\_input

course\_catalogue =[]

def check\_prequisite(course\_prerequisite):

"""Checking if the prerequisite exist or not"""

for course in course\_catalogue:

if course\_prerequisite in course.information['course\_code']:

return True

return False

def add\_course ():

"""To add new course in the course catalogue list"""

print ("\n\*\*\* ADD NEW COURSE \*\*\*")

course\_code = input ("\nEnter Course Code : ")

course\_title = input ("Enter Course Name : ")

course\_credit = input ("Enter Course Credit Hour : ")

course\_prerequisite = input ("Enter Course Prerequisite : ")

if course\_prerequisite == 'N/A':

course\_catalogue.append (Course(course\_code,course\_title,course\_credit,course\_prerequisite))

print ("\n\*\*\* Course Added \*\*\*")

elif check\_prequisite(course\_prerequisite):

course = Course(course\_code,course\_title,course\_credit,course\_prerequisite)

course\_catalogue.append(course)

print ("\n\*\*\* Course Added \*\*\*")

else:

print ("\n\*\*\* Course Prerequisite Doesn't Exist, Add It First \*\*\*")

add\_course()

def update\_course ():

"""Update Course Information"""

print ("\n\*\*\* UPDATE COURSE \*\*\*")

found = False

course\_code = input ("\nSearch the Course You want To Update, Search With Course Code: ")

for course in course\_catalogue:

if course\_code in course.information['course\_code']:

found = True

print ("\n\*\*\* Course Found! Course Information \*\*\*\n")

print (f"Course Code: {course.information['course\_code']}")

print (f"Course Name: {course.information['course\_title']}")

print (f"Course Credit Hour: {course.information['course\_credit']}")

print (f"Course Prerequisite: {course.information['course\_prerequisite']}\n")

print ("\n\*\*\* UPDATE INFORMATION \*\*\*\n")

course.information['course\_title']=input("Enter New Course Name: ")

course.information['course\_credit']=input("Enter New Credit Hour: ")

course.information['course\_prerequisite']=input("Enter New Prerequisite: ")

print ("\n\*\*\* Course Updated \*\*\*")

if(found==False):

print("\n\*\*\* Sorry, Course doesn't exist \*\*\*")

def delete\_course ():

"""Deleteing Course From Course Catalogue"""

print ("\n\*\*\* DELETE COURSE \*\*\*")

found = False

course\_code = input ("\nEnter Course Code of The Course To Be Deleted: ")

for course in course\_catalogue:

if course\_code in course.information['course\_code']:

found = True

print ("\n\*\*\* Course Found! Course Information \*\*\*\n")

print (f"Course Code: {course.information['course\_code']}")

print (f"Course Name: {course.information['course\_title']}")

print (f"Course Credit Hour: {course.information['course\_credit']}")

print (f"Course Prerequisite: {course.information['course\_prerequisite']}\n")

user\_input\_delete = input("Do You Want To Delete This Course, Press y To Delete The Course: ")

if (user\_input\_delete == "y"):

course\_catalogue.remove (course)

print ("\n\*\*\* Course Deleted \*\*\*")

if(found==False):

print("\n\*\*\* Sorry, Course doesn't exist \*\*\*")

def show\_all\_course ():

"""Display Information of All Course"""

if len(course\_catalogue)== 0:

print ("\n\*\*\* Nothing To Show, Add Course \*\*\*")

else:

print ("\n\*\*\* Course Catalogue \*\*\*\n")

for i in range(len(course\_catalogue)):

print (f"Course Code: {course\_catalogue[i].information['course\_code']}")

print (f"Course Name: {course\_catalogue[i].information['course\_title']}")

print (f"Course Credit Hour: {course\_catalogue[i].information['course\_credit']}")

print (f"Course Prerequisite: {course\_catalogue[i].information['course\_prerequisite']}\n")

def search\_course ():

"""Search Course And Displays Information If Found"""

print ("\n\*\*\* SEARCH COURSE \*\*\*")

found = False

course\_code = input ("\nEnter Course Code To Search: ")

for course in course\_catalogue:

if course\_code in course.information['course\_code']:

found = True

print ("\n\*\*\* Course Found! Course Information \*\*\*\n")

print (f"Course Code: {course.information['course\_code']}")

print (f"Course Name: {course.information['course\_title']}")

print (f"Course Credit Hour: {course.information['course\_credit']}")

print (f"Course Prerequisite: {course.information['course\_prerequisite']}\n")

if(found==False):

print("\n\*\*\* Sorry, Course doesn't exist \*\*\*")

add\_course ()

def store\_course\_to\_text ():

"""Store Course From System To A Text File"""

file\_name="course\_catalogue.txt"

with open(file\_name,'w') as file\_obj:

for i in range(len(course\_catalogue)):

file\_obj.writelines("\n")

file\_obj.writelines(f"Course Code: {course\_catalogue[i].information['course\_code']}\n")

file\_obj.writelines(f"Course Name: {course\_catalogue[i].information['course\_title']}\n")

file\_obj.writelines(f"Course Credit Hour: {course\_catalogue[i].information['course\_credit']}\n")

file\_obj.writelines(f"Course Prerequisite: {course\_catalogue[i].information['course\_prerequisite']}\n")

print ("\n\*\*\* Course Information Stored \*\*\*")

user\_interface\_input = user\_interface ()

while(user\_interface\_input != "quit"):

if(user\_interface\_input == "1"):

add\_course()

user\_interface\_input = user\_interface ()

elif(user\_interface\_input == "2"):

update\_course()

user\_interface\_input = user\_interface ()

elif(user\_interface\_input == "3"):

delete\_course()

user\_interface\_input = user\_interface ()

elif(user\_interface\_input == "4"):

show\_all\_course()

user\_interface\_input = user\_interface ()

elif(user\_interface\_input == "5"):

search\_course()

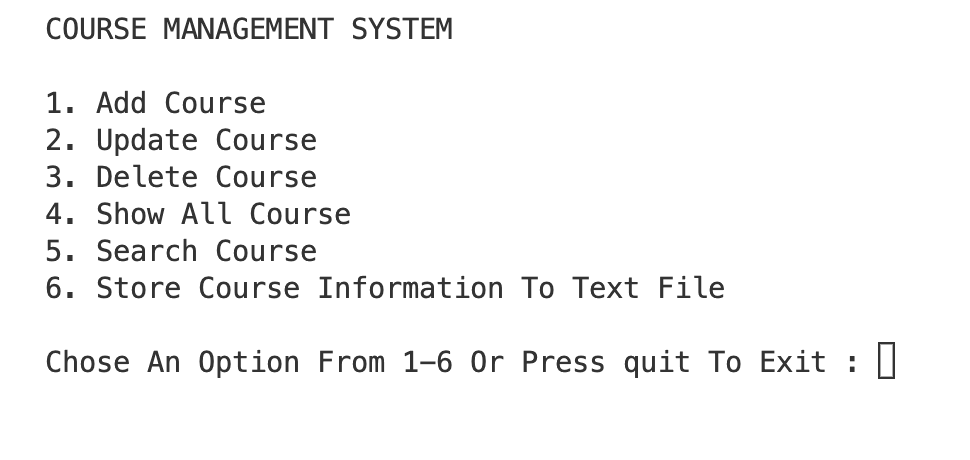
user\_interface\_input = user\_interface ()

elif(user\_interface\_input == "6"):

store\_course\_to\_text()

user\_interface\_input = user\_interface ()

**Project Outcomes:** When we run the application Course Management System Menu is displayed by user interface function. It offers user six options to choose from and can exit the system if “quite” is typed.



When the user choose option 1 from the menu add course function is prompted and it takes four inputs such as course code, course name, course credit hour and course prerequisite/s from the user. After user provide all information if the course prerequisite is of existing course code or “N/A” a confirmation message “Course Added” is displayed. Else user is requested to add the course first and add course is prompted from the beginning.

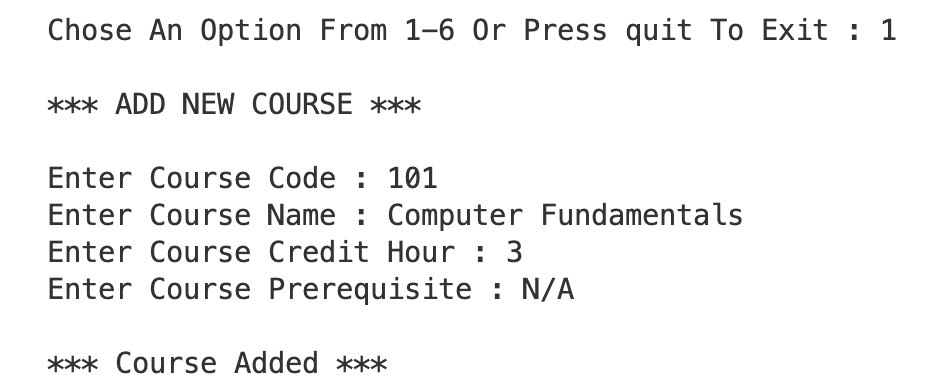


Fig: Adding course with no prerequisite

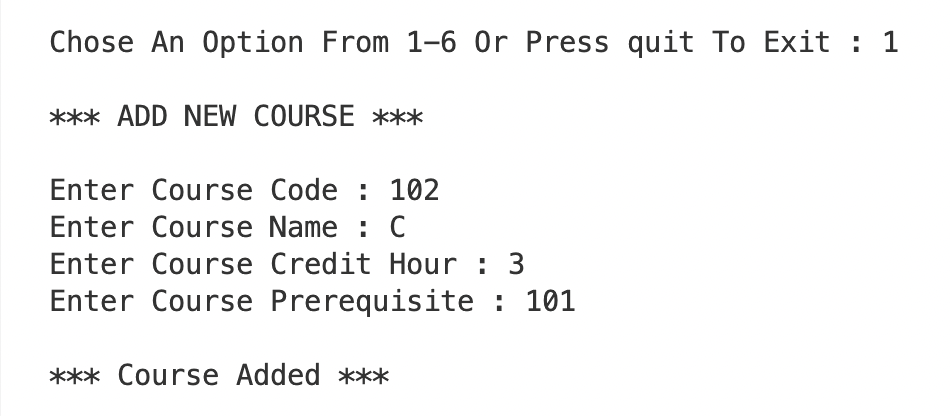


Fig: Adding course with existing course for prerequisite

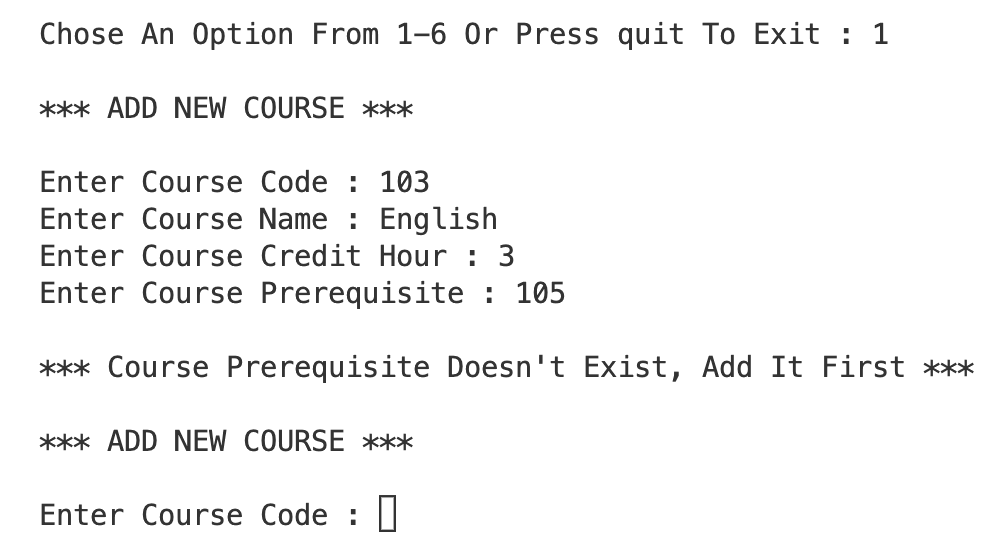


Fig: Adding course with non-existing course for prerequisite

If the user wants to update any course information, he chooses 2 and update course function is prompted. It first asks the user to search the course he wants to update with the course code. If the course is found it gives a confirmation message and displays the course information. Then user input is taken such as course name, course credit and course prerequisite in order to overwrite the previous information, once it is done it displays a

“Course Updated” message, in case the course doesn’t exist it shows an error message “Sorry, Course doesn't exist”.

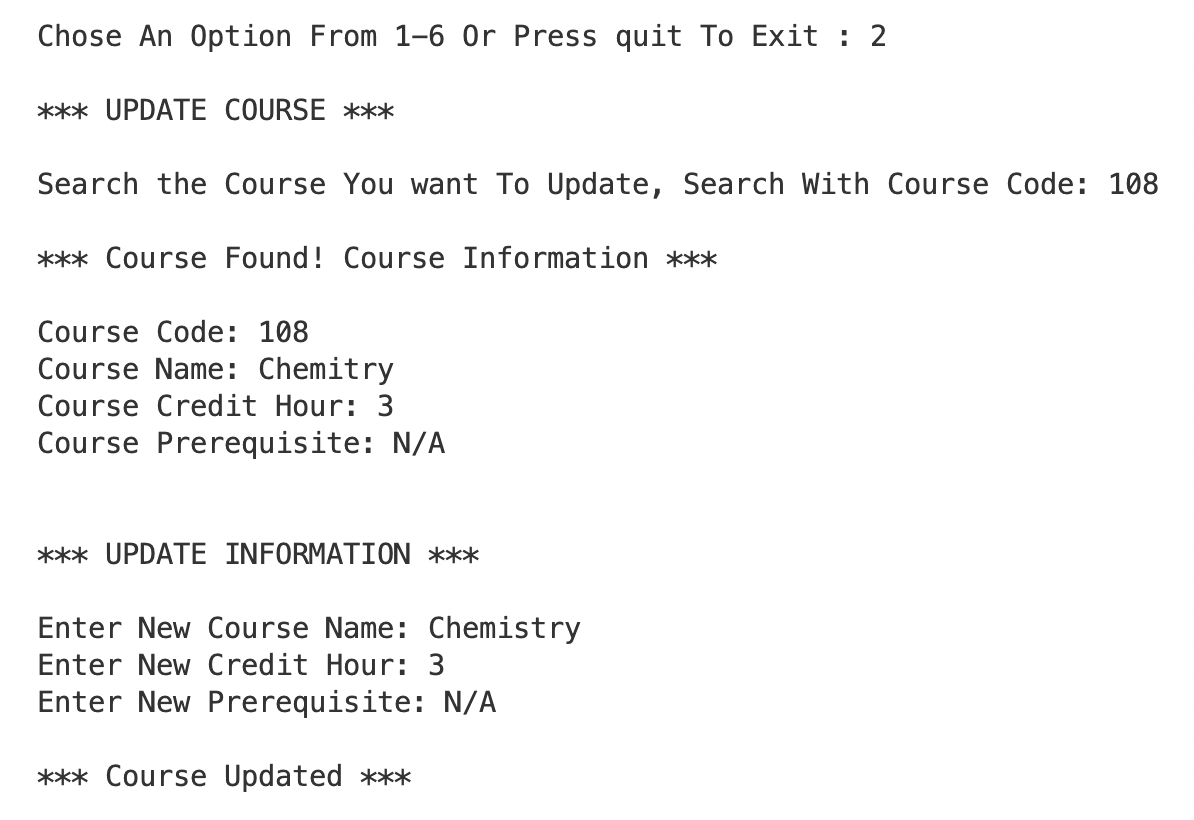


Fig: Updating Course Information

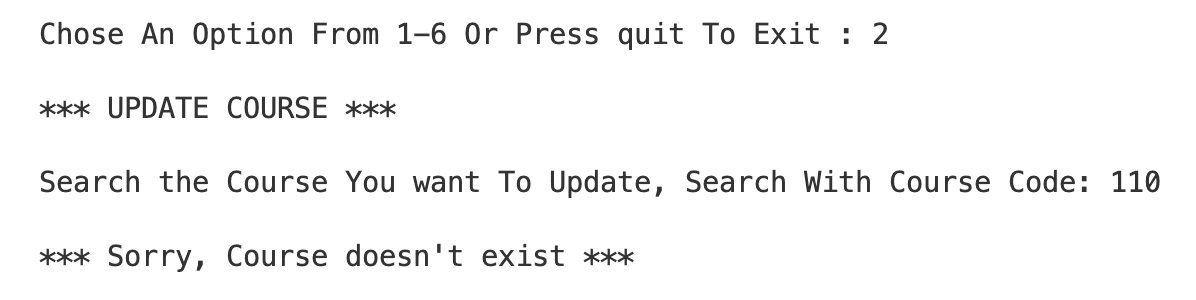


Fig: If the course is not found for updating

In option 3 user can delete course, delete course function is initiated and it ask user to enter the course code of the course he wants to delete, if the course is found it shows course found message and displays the course’s detailed in formation and it takes a confirmation from the user and “Course Deleted” message is shown once the course is deleted.

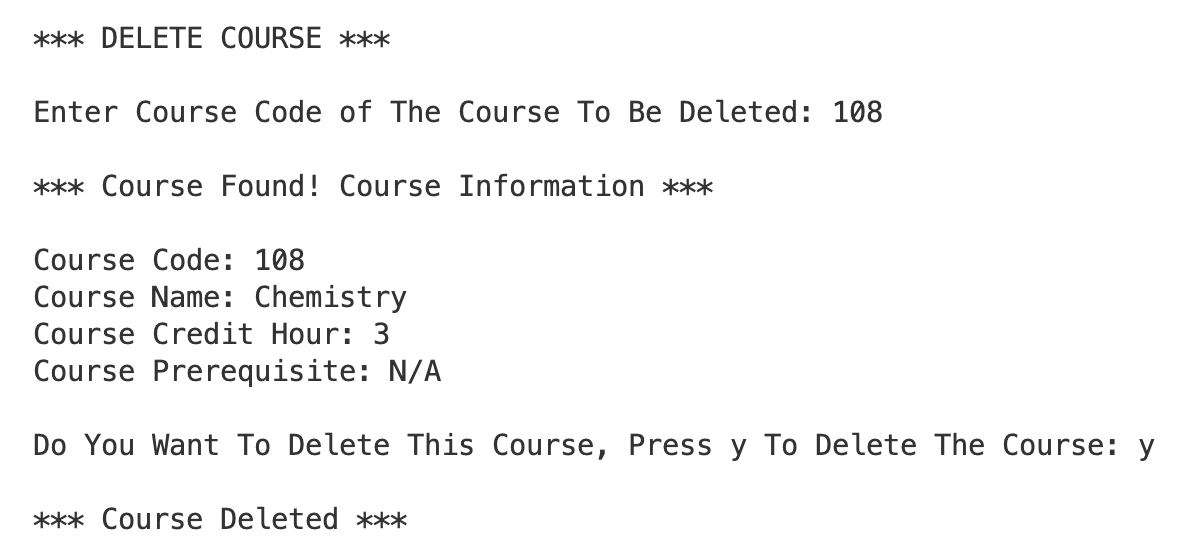


Fig: Deleting Course

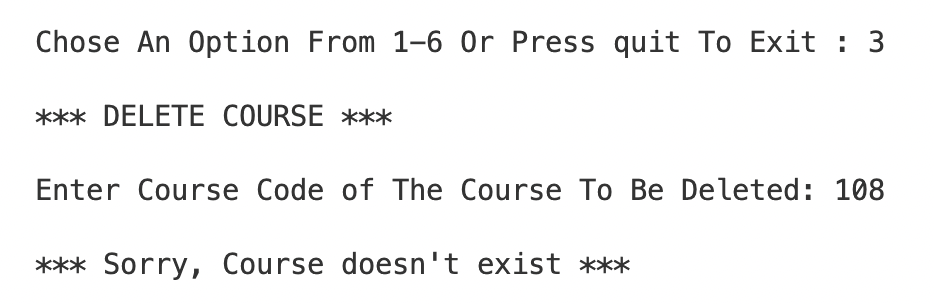


Fig: If the course doesn’t exist for deleting

In the option 4, the user can check all course in the course catalogue and their detailed information unless the catalogue is empty. For this show all course function is used.

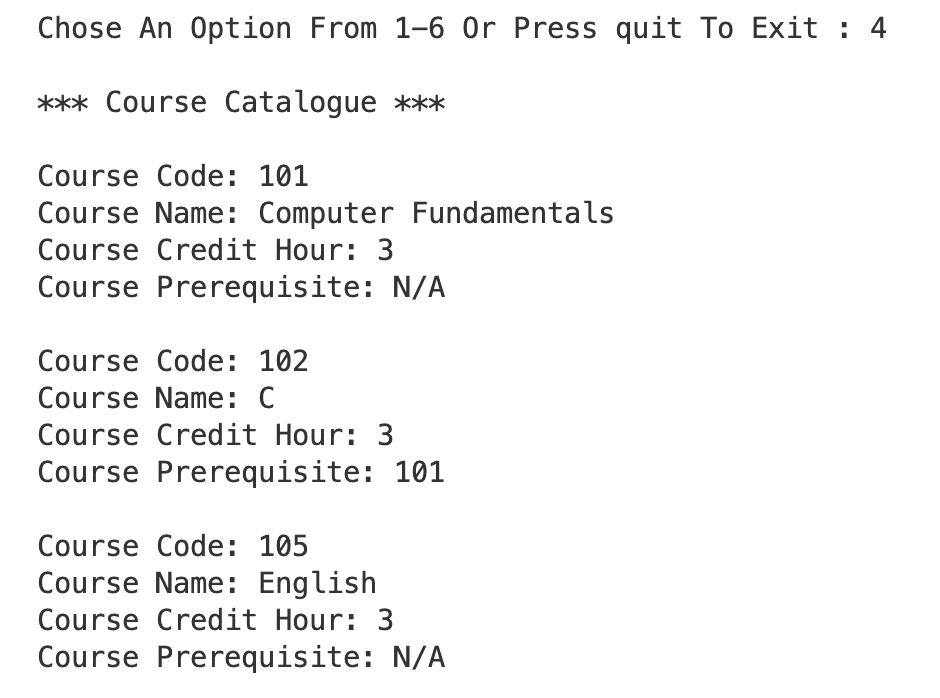


Fig: Show all course

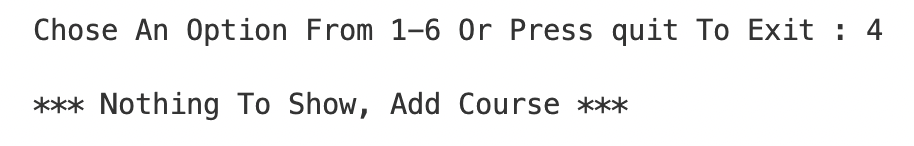


Fig: Empty Course Catalogue

Option 5 is search course option, search course function is used for this feature, it asks user the course they want to search and user gives the course code as the input. If the course is found “Course Found! Course Information” message is displayed along with the detailed information of that course. If the course code doesn’t exist it prompt add course function following error message.

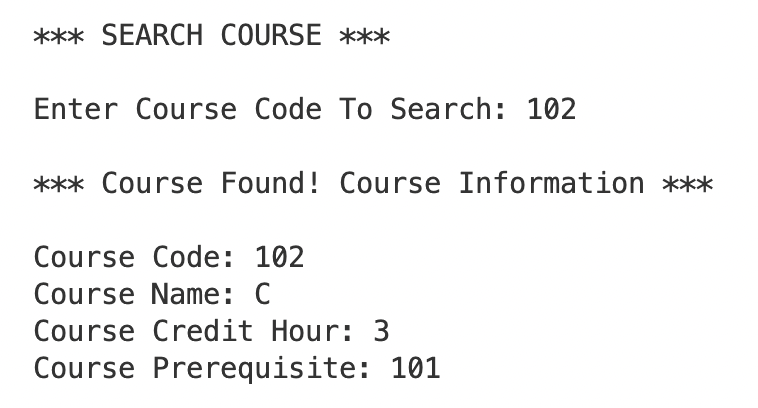


Fig: Search Course

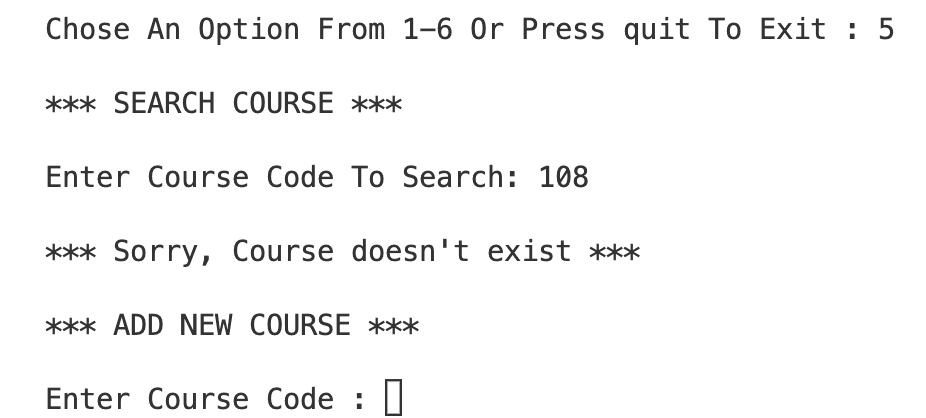


Fig: If the course doesn’t exist upon searching

If the user chose option 6, they can store the courses from the course catalogue to a text file so they can have a physical copy for any further use. Store course to text function is used to implement this feature. After the course is store a conformation message “Course Information Stored” is displayed.

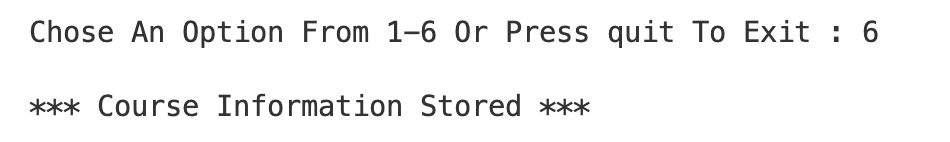


Fig: Storing Course into Text File

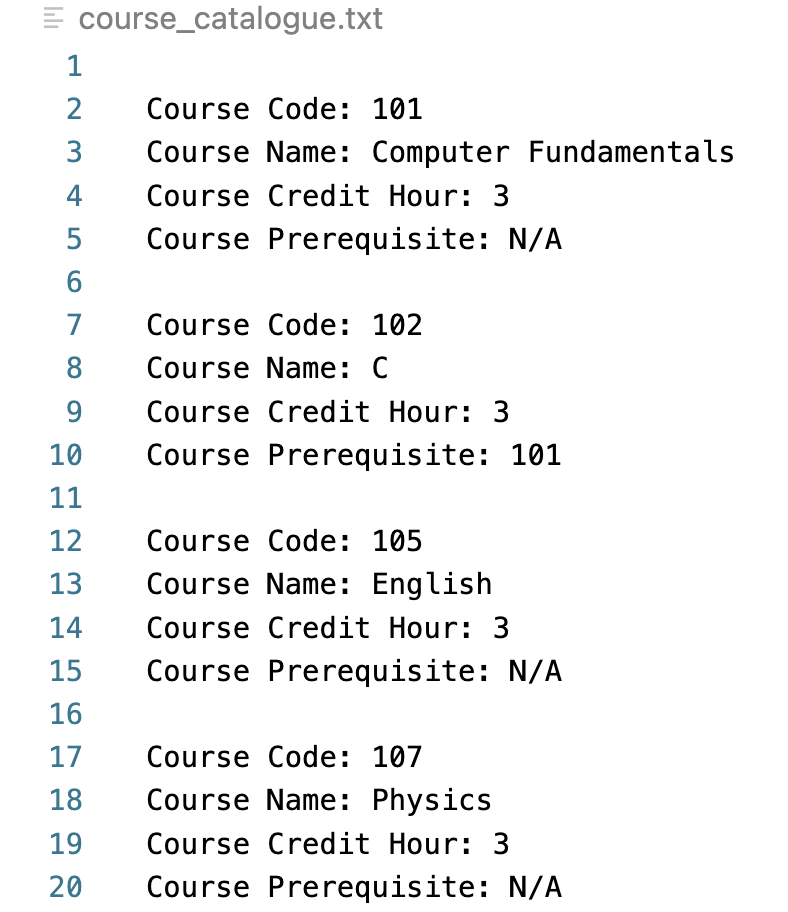


Fig: Text File

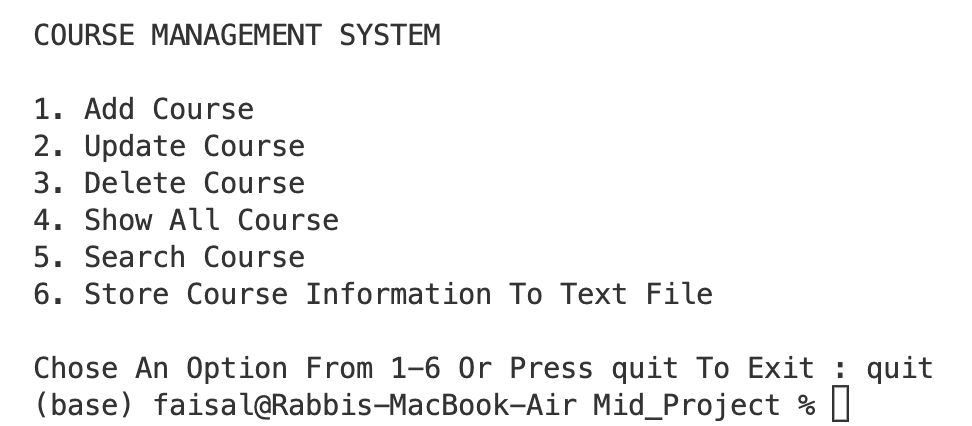


Fig: If user input is “quit”