Kingdom of Saudi Arabia
Ministry of Education
Imam Abdulrahman bin Faisal University
College of Science and Humanities-Jubail
Department Computer Science

المملكة العربية السعودية
وزارة التعليم
جامعة الإمام عبدالرحمن بن فيصل
كلية العلوم والدراسات الإنسانية-الجبيل
قسم علوم الحاسب

CS 516 Advanced Programming Language (Term 3, 2022-2023)

Neat Notes

Report Submitted to:

Dr. Huda Althumali & Mrs. Maha Alghamdi

Kingdom of Saudi Arabia
Ministry of Education
Imam Abdulrahman bin Faisal University
College of Science and Humanities-Jubail
Department Computer Science

المملكة العربية السعودية
وزارة التعليم
جامعة الإمام عبدالرحمن بن فيصل
كلية العلوم والدراسات الإنسانية-الجبيل
قسم علوم الحاسب

Neat Notes

Declaration Statement:

This work is dedicated to Imam Abdulrahman bin Faisal University, Collage of science and humanities - Computer Science - department. Based on Advanced Programming Language (CS 516) course requirements.

Submitted by:

Name: Danah Habeeb Almuzel	Student ID: 2190001201
Name: Reem Shaker Almualem	Student ID: 2190000429
Name: Areej Ahmed Saleh	Student ID: 2190006001
Name: Lulu Waleed Aldhwaihi	Student ID: 2190004563
Name: Hana Ali Alomran	Student ID: 2190004931

Contents

Table of Figures

List of Tables

1. Problem Statement:

University students always choose to share their notes and use test banks during studying for quizzes, midterms, and finals. However, students occasionally have trouble locating the questions they shared to review since an excessive number of questions leads to their becoming mixed up with other files and messages. In Neat Notes system, the system is managed by the admin and the student will be able to view the questions and notes of the chosen course.

2. Project Goals and Objectives/Deliverables:

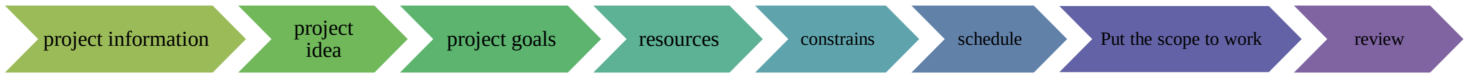
- Users can log in to their accounts easily.
- Users can add, delete, update, and view questions.
- Users can create and name a file for each course.
- Users can post their notes for any course.
- Users can add and view their notes, self-tests, prior exams, and more.
- Users can search for a specific course to find the desired notes.

3. Introduction:

The system Neat Notes has been implemented using two programming languages, Java and C, and a comparison has been made between the procedural paradigm and the object-oriented paradigm to highlight their differences. Neat Notes system asset university students to share notes and access test banks when preparing for exams, Neat Notes gathers notes and questions all in one place to help students not get overwhelmed finding and locating where the notes and questions are when trying to review. Neat Notes got two main users, firstly, the administrator who is responsible for adding and deleting courses, as well as adding add various types of questions (e.g., multiple choice questions and fill-in-the-blank questions) for each added course in the system. Secondly, the student can enroll in the added courses and view the questions for the courses the student is enrolled in. Neat Notes system has been implemented using two programming languages, Java and C, and a comparison has been made between the procedural paradigm and the object-oriented paradigm to highlight their differences.

4. Project Scope:

- Collect project information.
- Choose project idea.
- Define goals and objectives/deliverables.
- Assess available resources to choose two programming languages based on it.
- Consider the constrains limitations/restrictions.
- Build out a schedule and deadlines.
- Put the scope to work and start coding and filling out the project form before the actual deadline.
- Final review and proofread before the submission.



5. Success Factors and Benefits:

- ***More comfort***:

Users will be able to enter their accounts and conveniently access their notes, self-tests, previous exams, and other relevant information.

- ***Better user experience:***

The system effectively enhances user experience by allowing users to add, delete, update, and view questions as well as create and name a file for each course.

- ***Time-saving:***

By enabling users to post their notes for any course, they can share their notes with their peers for better studying and avoid constantly retyping their notes on every device they use. In addition, that the user can easily search by entering the course ID for the program to display the files related to it.

6. Limitations/Restrictions:

- ***Inappropriate use:***

Concerns about using the system improperly, such as uploading inappropriate content, the attempt to hack the system, and disclosing other users' private information.

- ***User Error:***

There is a chance that user-generated content will include errors or inconsistencies that affect the course notes' accuracy or consistency.

- ***System updates:***

Due to the absence of databases in the system, change/s upgrades may cause the loss of some data.

- ***Interfaces:***

The system lacks interfaces, which could make things quite confusing for those who aren't familiar with it.

7. Selected programming language 1:

The first selected programming language is Java language. Sun Microsystems created the Java programming language in the early 1990s. It is a high-level, portable, object-oriented programming language. It is typically used for web-based applications. Java is both a compiled and an interpreted language since its source code is first compiled into binary byte code before being executed by the interpreter at run time. This byte code executes on the Java Virtual Machine (JVM), a software-based interpreter. This byte code is responsible for Java's portability: it will execute on any suitably implemented JVM, independent of machine hardware or software configuration [1][2]. Some of Java's programming language strengths, limitations, and features are: [2]

Strengths:

- Platform independence: Java is a highly portable language that can be built and used on any platform that has a Java Virtual Machine (JVM).
- is a pure object-oriented programming language, which implies that encapsulation, inheritance, and polymorphism are all supported by the language by default.
- Memory management is made simple, and the likelihood of memory leaks and other memory-related issues is decreased thanks to Java's automated memory management through garbage collection.

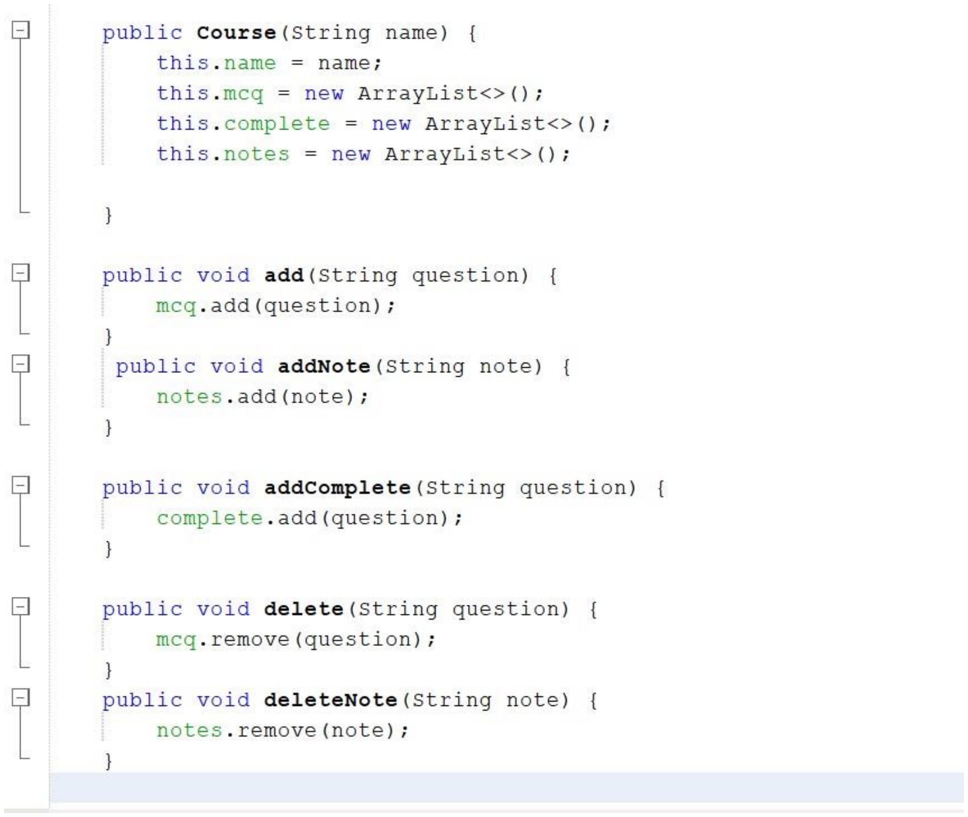
Limitations:

- applications that need low-level access to computer hardware, Java may perform slower than alternative languages like C or C++.
- Java is a complicated language that may be challenging to master, especially for newcomers.
- Limited access to computer hardware: Java only offers a limited amount of access to the hardware, which can make it unsuitable for some applications that need fine-grained hardware control.

Features:

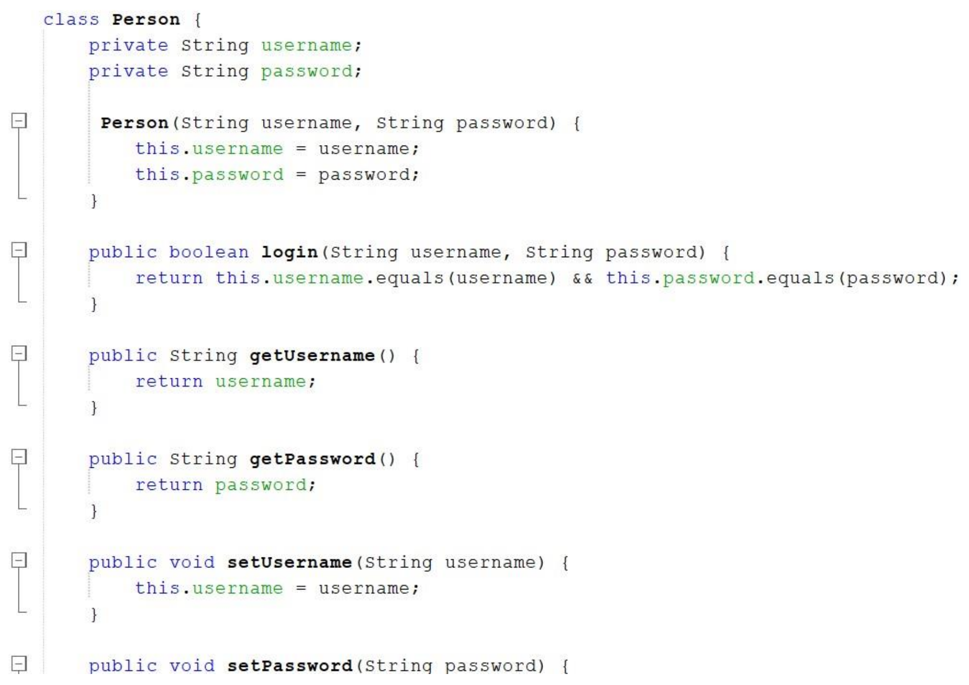
- Java offers a variety of APIs that span a wide spectrum of capabilities, from databases and GUI development to networking and I/O.
- Java comes with built-in multi-threading capabilities, enabling developers to create concurrent applications that can make use of multi-core CPUs.
- Protection against malicious code and other security risks is made possible by Java's strong security capabilities, which include a security manager and a sandboxed environment.

Java language was implemented to create Neats Notes system. This system contains three main different classes, Courses, Editor, and Student. Since the Editor class and Student class have common properties, a superclass was created to inherit its properties to the subclasses as shown in the figures below:



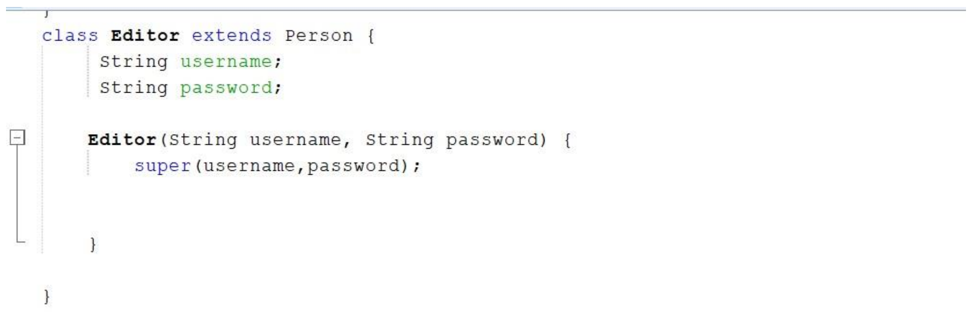
```
public Course(String name) {
    this.name = name;
    this.mcq = new ArrayList<>();
    this.complete = new ArrayList<>();
    this.notes = new ArrayList<>();

}

public void add(String question) {
    mcq.add(question);
}
 public void addNote(String note) {
    notes.add(note);
}

public void addComplete(String question) {
    complete.add(question);
}

public void delete(String question) {
    mcq.remove(question);
}
public void deleteNote(String note) {
    notes.remove(note);
}
```

Figure 1 Course Class



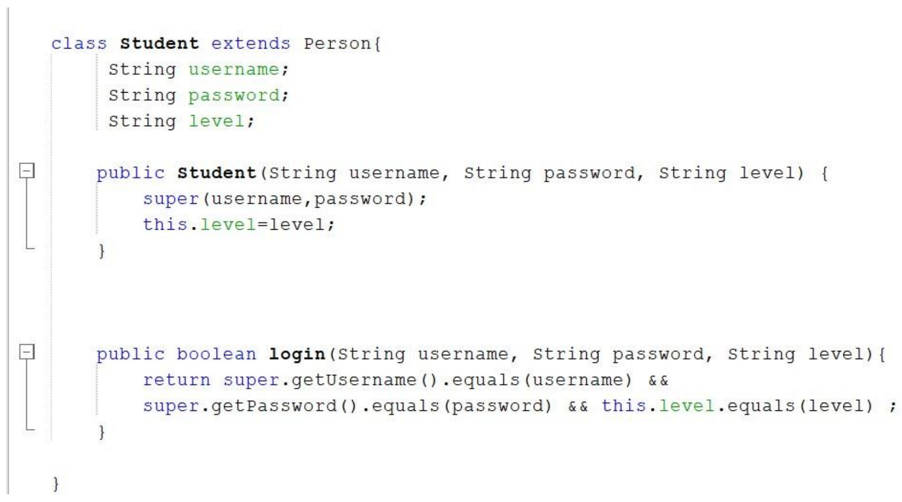
```java
class Person {
    private String username;
    private String password;

     Person(String username, String password) {
        this.username = username;
        this.password = password;
    }

    public boolean login(String username, String password) {
        return this.username.equals(username) && this.password.equals(password);
    }

    public String getUsername() {
        return username;
    }

    public String getPassword() {
        return password;
    }

    public void setUsername(String username) {
        this.username = username;
    }

    public void setPassword(String password) {
```

Figure 2 Person Class



```java
class Editor extends Person {
     String username;
     String password;

    Editor(String username, String password) {
        super(username,password);


    }

}
```

Figure 3 Editor class



```java
class Student extends Person{
     String username;
     String password;
     String level;

    public Student(String username, String password, String level) {
        super(username,password);
        this.level=level;
    }



    public boolean login(String username, String password, String level){
        return super.getUsername().equals(username) &&
        super.getPassword().equals(password) && this.level.equals(level) ;
    }

}
```

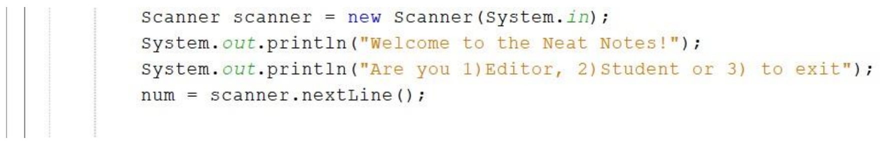
Figure 4 Student class

This system has many functionalities and features. In this section, all system's functionalities and features will be discussed and shown:

- **Welcome to the system.**

When the users enter the system, three options will be shown so the users can choose from. As seen in the figures below:

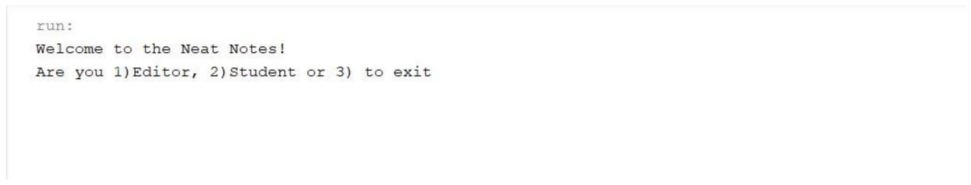
The code:



```
Scanner scanner = new Scanner(System.in);
System.out.println("Welcome to the Neat Notes!");
System.out.println("Are you 1)Editor, 2)Student or 3) to exit");
num = scanner.nextLine();
```

Figure 5 welcome code

The corresponding output:



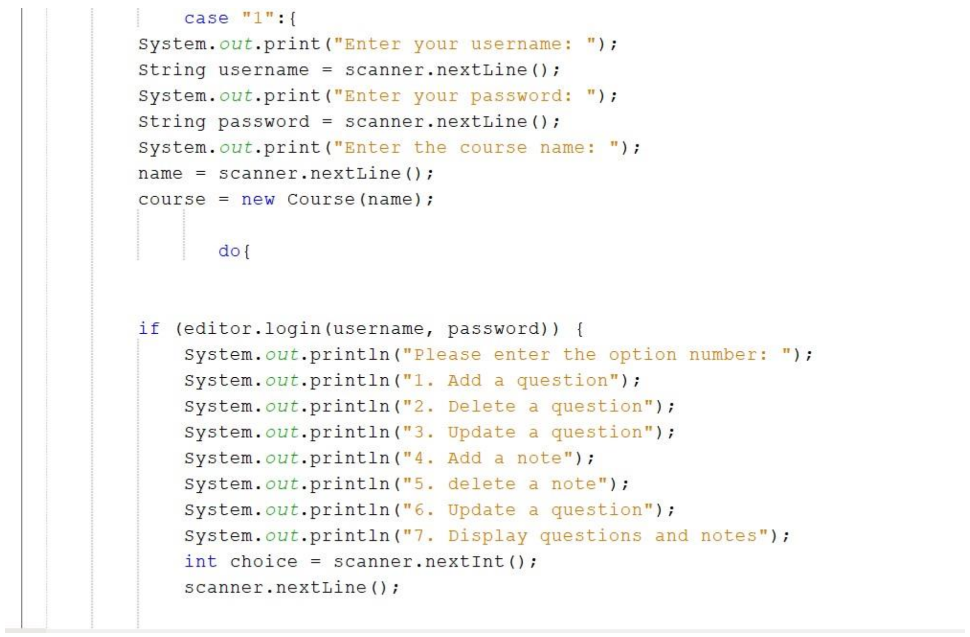
```
run:
Welcome to the Neat Notes!
Are you 1)Editor, 2)Student or 3) to exit
```

Figure 6 welcome output

- **Editor's section**

When the user is an editor, she/he should type 1, and log in to the system, and then the editor's functionalities will appear to allow the editor to choose one option. As seen in the following figures:

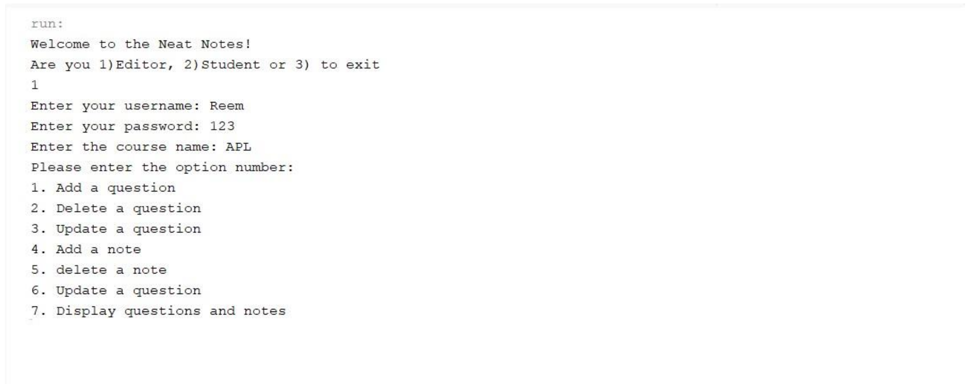
The code:



```
        case "1":{
System.out.print("Enter your username: ");
String username = scanner.nextLine();
System.out.print("Enter your password: ");
String password = scanner.nextLine();
System.out.print("Enter the course name: ");
name = scanner.nextLine();
course = new Course(name);

        do{


if (editor.login(username, password)) {
    System.out.println("Please enter the option number: ");
    System.out.println("1. Add a question");
    System.out.println("2. Delete a question");
    System.out.println("3. Update a question");
    System.out.println("4. Add a note");
    System.out.println("5. delete a note");
    System.out.println("6. Update a question");
    System.out.println("7. Display questions and notes");
    int choice = scanner.nextInt();
    scanner.nextLine();
```

Figure 7 Editor code

The corresponding output:



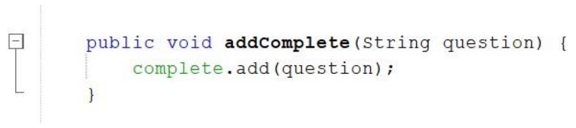
```
run:
Welcome to the Neat Notes!
Are you 1)Editor, 2)Student or 3) to exit
1
Enter your username: Reem
Enter your password: 123
Enter the course name: APL
Please enter the option number:
1. Add a question
2. Delete a question
3. Update a question
4. Add a note
5. delete a note
6. Update a question
7. Display questions and notes
```

Figure 8 Editor output

- **Adding a Complete question**

To add new questions the editor should type 1 and determine the type of question and the question itself. As seen in the following figures:

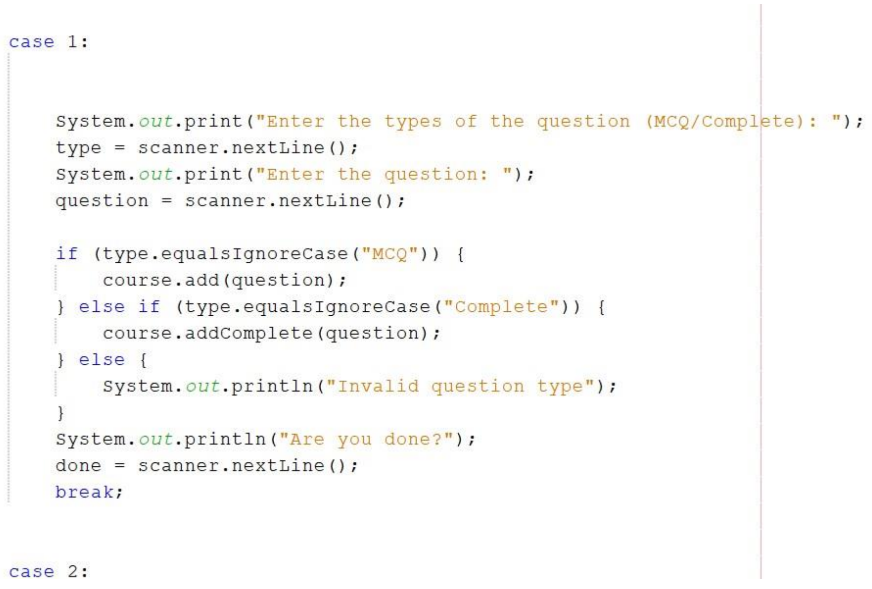
The code in course class:



```
public void addComplete(String question) {
    complete.add(question);
}
```

Figure 9 adding code in the class

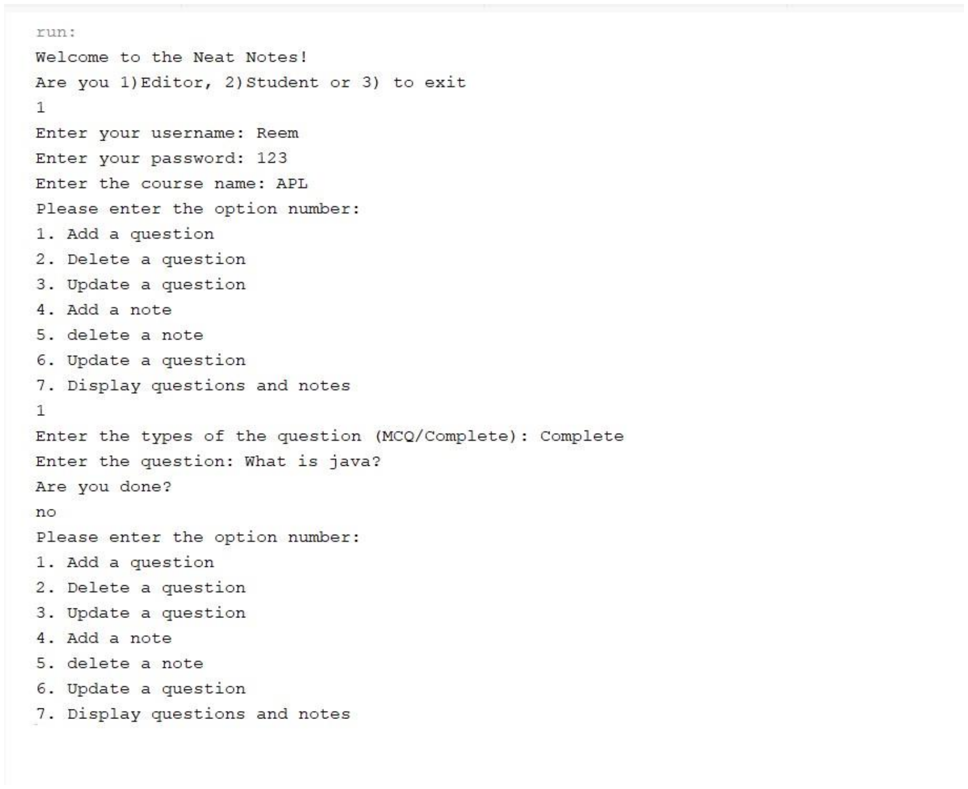
The code in main class:



```
case 1:


    System.out.print("Enter the types of the question (MCQ/Complete): ");
    type = scanner.nextLine();
    System.out.print("Enter the question: ");
    question = scanner.nextLine();

    if (type.equalsIgnoreCase("MCQ")) {
        course.add(question);
    } else if (type.equalsIgnoreCase("Complete")) {
        course.addComplete(question);
    } else {
        System.out.println("Invalid question type");
    }
    System.out.println("Are you done?");
    done = scanner.nextLine();
    break;


case 2:
```

Figure 10 adding code in main class

The corresponding output:



```
run:
Welcome to the Neat Notes!
Are you 1)Editor, 2)Student or 3) to exit
1
Enter your username: Reem
Enter your password: 123
Enter the course name: APL
Please enter the option number:
1. Add a question
2. Delete a question
3. Update a question
4. Add a note
5. delete a note
6. Update a question
7. Display questions and notes
1
Enter the types of the question (MCQ/Complete): Complete
Enter the question: What is java?
Are you done?
no
Please enter the option number:
1. Add a question
2. Delete a question
3. Update a question
4. Add a note
5. delete a note
6. Update a question
7. Display questions and notes
```

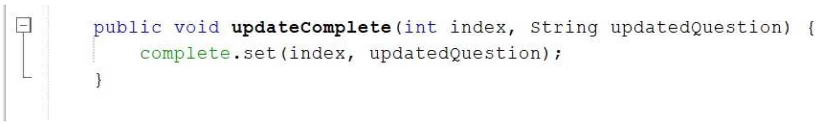
Figure 11 output of adding a complete question

The editors will be asked again whether they are finished or want to complete using the system. When the editor types no, all system functionalities will appear again as seen in the previous figure.

- **Updating a question**

When the editor wants to update the added question, she/he should type 3, the added questions will be displayed, and the editor will be asked about the type of question to be updated and the index of it and then the editor should type the new updated question.

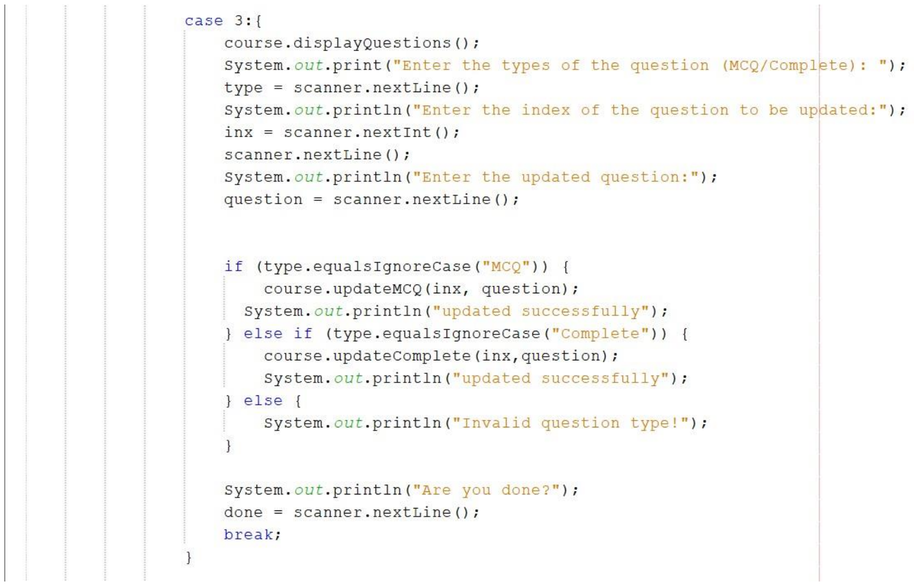
The code in Course class:



```java
public void updateComplete(int index, String updatedQuestion) {
    complete.set(index, updatedQuestion);
}
```

Figure 12 updating a question code in Course class

The code in main class:



```java
case 3:{
    course.displayQuestions();
    System.out.print("Enter the types of the question (MCQ/Complete): ");
    type = scanner.nextLine();
    System.out.println("Enter the index of the question to be updated:");
    inx = scanner.nextInt();
    scanner.nextLine();
    System.out.println("Enter the updated question:");
    question = scanner.nextLine();


    if (type.equalsIgnoreCase("MCQ")) {
        course.updateMCQ(inx, question);
      System.out.println("updated successfully");
    } else if (type.equalsIgnoreCase("Complete")) {
        course.updateComplete(inx,question);
        System.out.println("updated successfully");
    } else {
        System.out.println("Invalid question type!");
    }

    System.out.println("Are you done?");
    done = scanner.nextLine();
    break;
}
```

Figure 13 updating a question code in main class

The corresponding output:

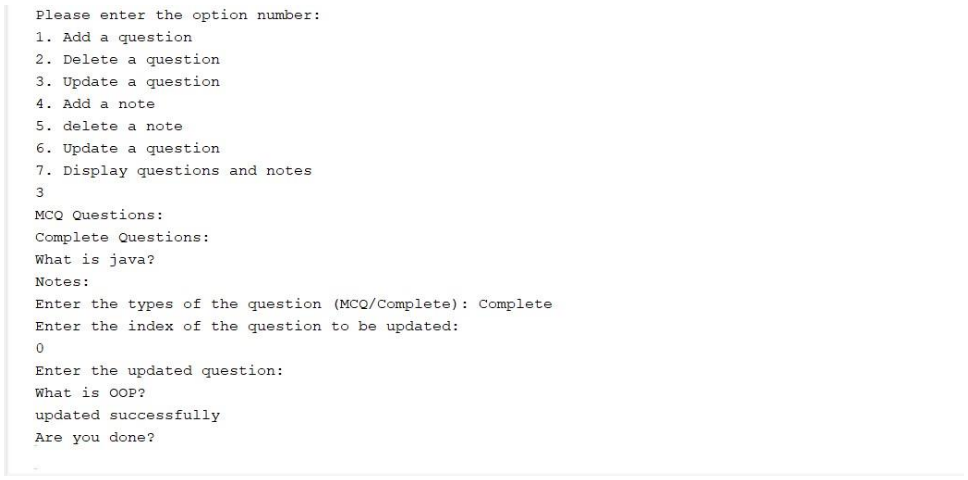

```
Please enter the option number:
1. Add a question
2. Delete a question
3. Update a question
4. Add a note
5. delete a note
6. Update a question
7. Display questions and notes
3
MCQ Questions:
Complete Questions:
What is java?
Notes:
Enter the types of the question (MCQ/Complete): Complete
Enter the index of the question to be updated:
0
Enter the updated question:
What is OOP?
updated successfully
Are you done?
```

Figure 14 updating a question output

- **Deleting a question**

When the editor wants to delete a question, she/he should type 2, and then the editor will be asked about the type of question to be deleted, the last added question will be deleted

The code in Course class:

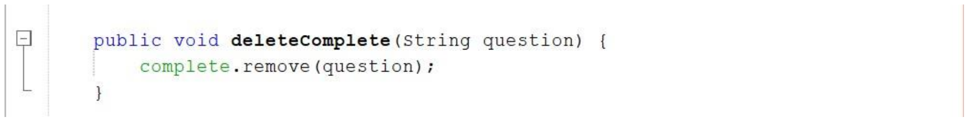

```
public void deleteComplete(String question) {
    complete.remove(question);
}
```

Figure 15 deleting code in the Course class

The Code in main class:

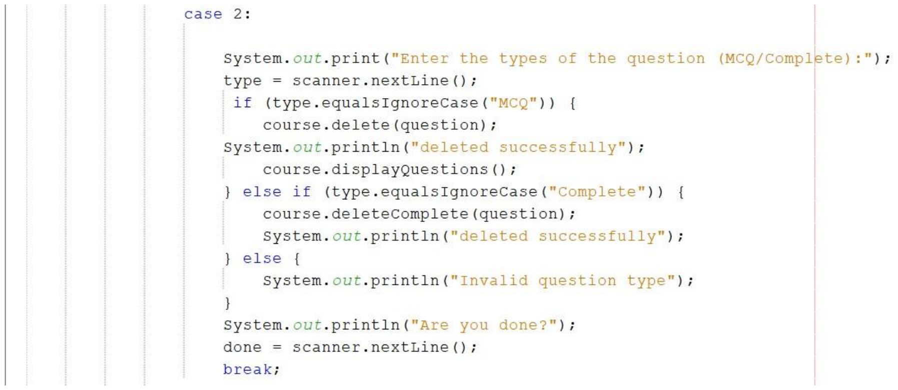

```
case 2:

    System.out.print("Enter the types of the question (MCQ/Complete):");
    type = scanner.nextLine();
     if (type.equalsIgnoreCase("MCQ")) {
        course.delete(question);
    System.out.println("deleted successfully");
        course.displayQuestions();
    } else if (type.equalsIgnoreCase("Complete")) {
        course.deleteComplete(question);
        System.out.println("deleted successfully");
    } else {
        System.out.println("Invalid question type");
    }
    System.out.println("Are you done?");
    done = scanner.nextLine();
    break;
```

Figure 16 deleting a question code in main class

The corresponding output:

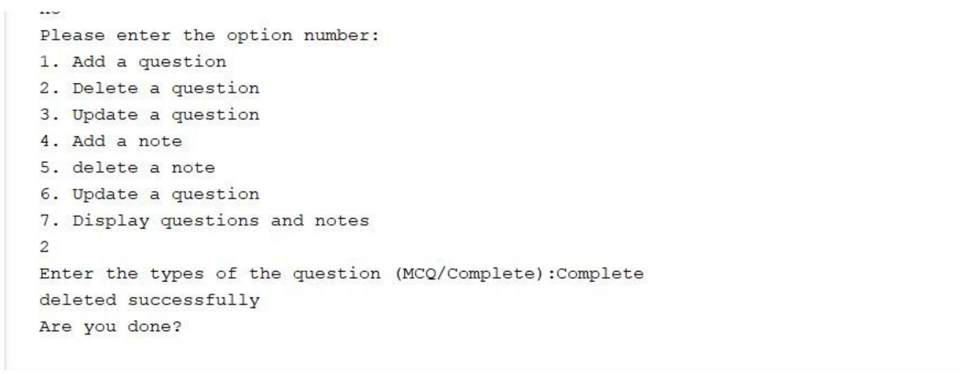

```
Please enter the option number:
1. Add a question
2. Delete a question
3. Update a question
4. Add a note
5. delete a note
6. Update a question
7. Display questions and notes
2
Enter the types of the question (MCQ/Complete):Complete
deleted successfully
Are you done?
```

Figure 17 deleting a question output

The remaining options: adding a note, deleting a note, and updating a note have the same functionalities as adding a question, deleting a question, and updating a question.

- **Student's Section**

When the user is a student, she/ he should type 2, to log in to the system the student should type the username, the password, and the level. Then the student should select the desired course to display the questions and the notes that were added by the editor.

The code of display method in the Course class:

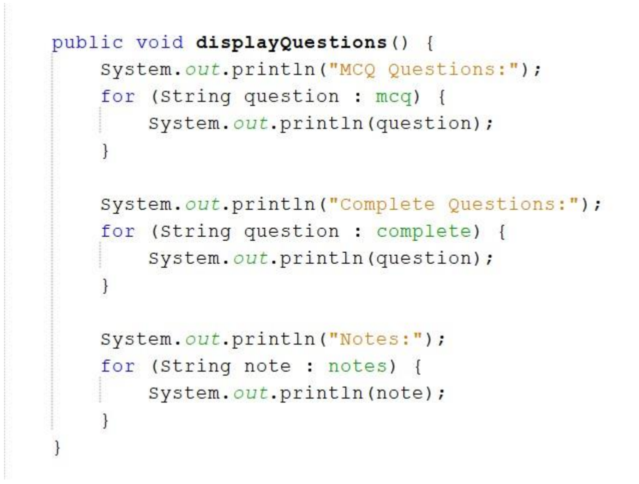
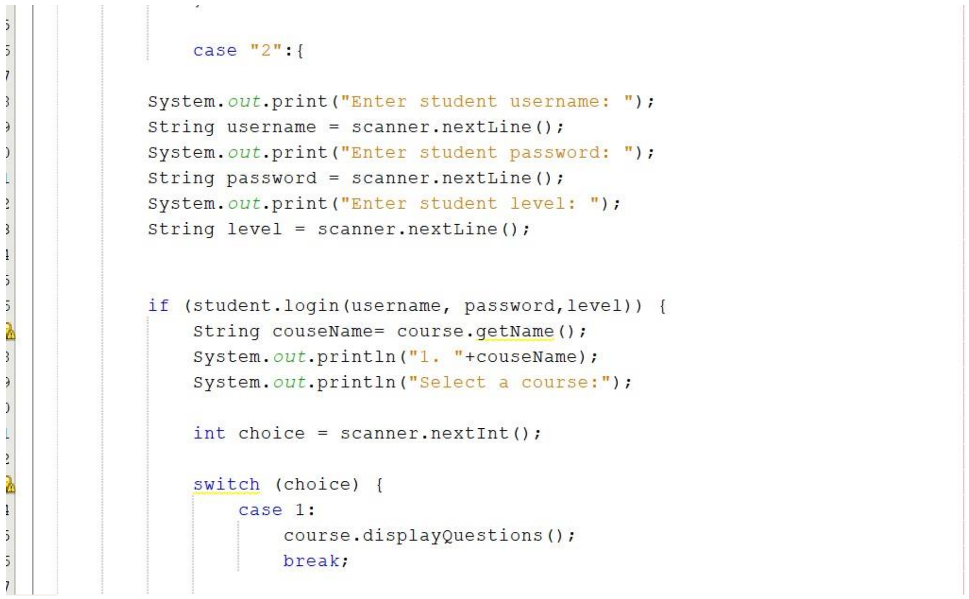

```java
public void displayQuestions() {
    System.out.println("MCQ Questions:");
    for (String question : mcq) {
        System.out.println(question);
    }

    System.out.println("Complete Questions:");
    for (String question : complete) {
        System.out.println(question);
    }

    System.out.println("Notes:");
    for (String note : notes) {
        System.out.println(note);
    }
}
```

Figure 18 Displaying questions and notes code in Course class

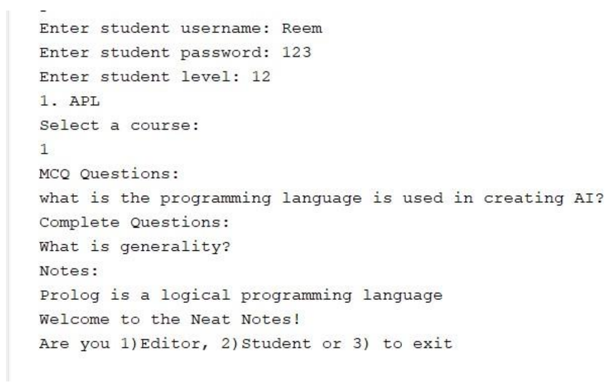
The code of student section in the main class:



```
        case "2":{

    System.out.print("Enter student username: ");
    String username = scanner.nextLine();
    System.out.print("Enter student password: ");
    String password = scanner.nextLine();
    System.out.print("Enter student level: ");
    String level = scanner.nextLine();


    if (student.login(username, password,level)) {
        String couseName= course.getName();
        System.out.println("1. "+couseName);
        System.out.println("Select a course:");

        int choice = scanner.nextInt();

        switch (choice) {
            case 1:
                course.displayQuestions();
                break;
```

Figure 19 Student log in and displaying questions and notes code in main class

The corresponding output:



```
-
Enter student username: Reem
Enter student password: 123
Enter student level: 12
1. APL
Select a course:
1
MCQ Questions:
what is the programming language is used in creating AI?
Complete Questions:
What is generality?
Notes:
Prolog is a logical programming language
Welcome to the Neat Notes!
Are you 1)Editor, 2)Student or 3) to exit
```

Figure 20 Student output

8. Selected programming language 2:

The second selected programming language is C language. Dennis Ritchie at Bell Labs created the general-purpose computer language C in the early 1970s. Compared to interpreted languages, it is a high-level language that is compiled into machine code, making it quicker and more effective. Although C has a limited number of keywords and syntax, it offers low-level functionality including bit-level operations, pointer manipulation, and direct memory access. There are several applications for C. Additionally, it influenced the creation of several other programming languages, including C++, Java, and Python. C's popularity is because of its portability, speed, and efficiency as well as its capability to communicate directly with hardware and operating systems, C is quite popular [2]. Some of C's programming language strengths, limitations, and features are: [2]

Strengths:

- The C programming language is quick and efficient and offers direct access to the hardware and memory of a computer, making it ideal for system-level programming and applications with a high priority on performance.
- C is a highly portable language since it can be compiled and run on a variety of systems. This is partly because there are C compilers and libraries available for a wide range of operating systems and hardware architectures.
- C has a strong collection of features that let programmers build sophisticated data structures and algorithms, including pointers, structures, and arrays. It is a popular option for creating operating systems, device drivers, and other low-level software because of its versatility.

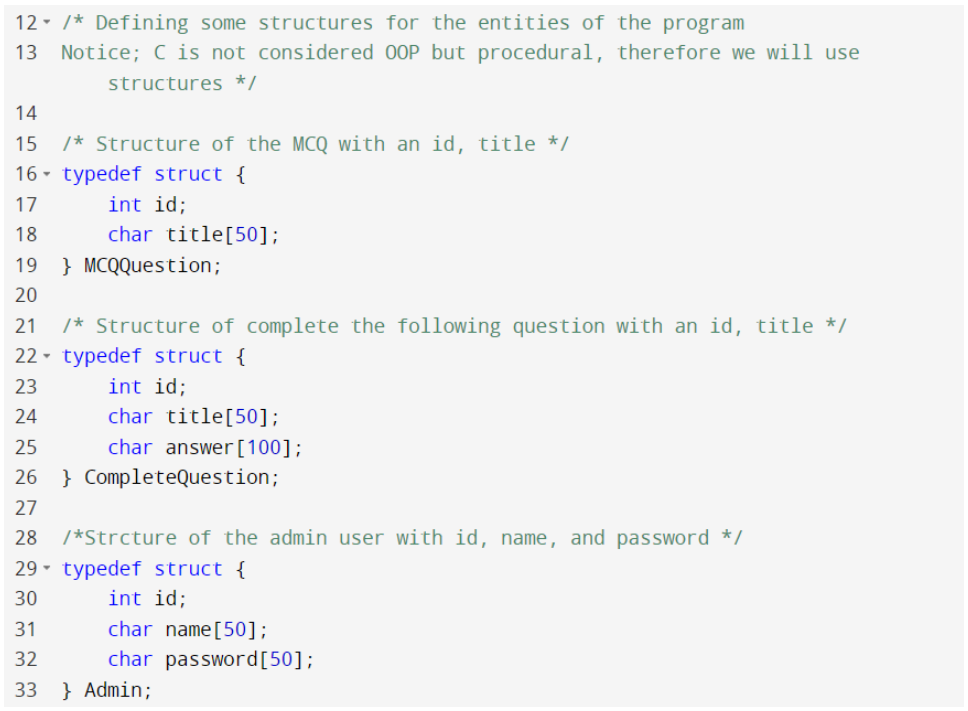
Limitations:

- Low-level programming: Because C is a low-level language, it does not offer many abstractions for higher-level ideas like object-oriented programming and needs manual memory management.
- Security flaws: Because C offers direct access to computer memory, it may be susceptible to flaws like memory leaks and buffer overflows. To prevent these problems, proper development techniques and security testing are needed.
- steep learning curve: Programmers accustomed to higher-level languages with greater abstractions and automated memory management may find it challenging to learn C

Features:

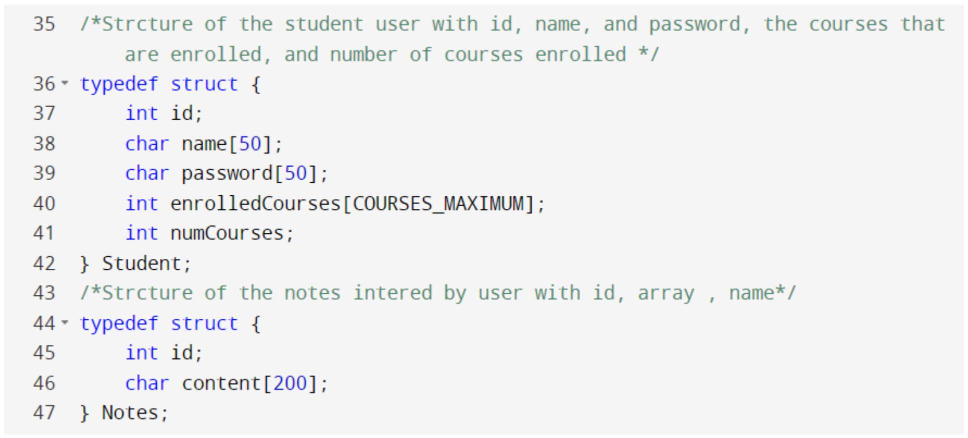
- Pointers: The strong feature of pointers that C offers lets programmers directly handle memory and build intricate data structures.
- Functions and loops are examples of structured programming principles that C offers, making it simpler to create and manage complex systems.
- Standard library: To make it simpler to build portable code, C comes with a standard library of functions and data types that may be used by any C application.

C language was also implemented to create Neat Notes system. The code corresponds to the Java code functionality but differs in syntax and dependencies. Since C language does not support classes inheritance the code was implemented using structures. In this code, the structure was used for the (MCQ Questions - Complete the following questions - Notes – Admin user – Student user - Course) with ID's, arrays, structures names and all features needed to fulfill the same functionality of the java code. The aim of using the structures is to group related data items and variables with different data types together in a single entity.



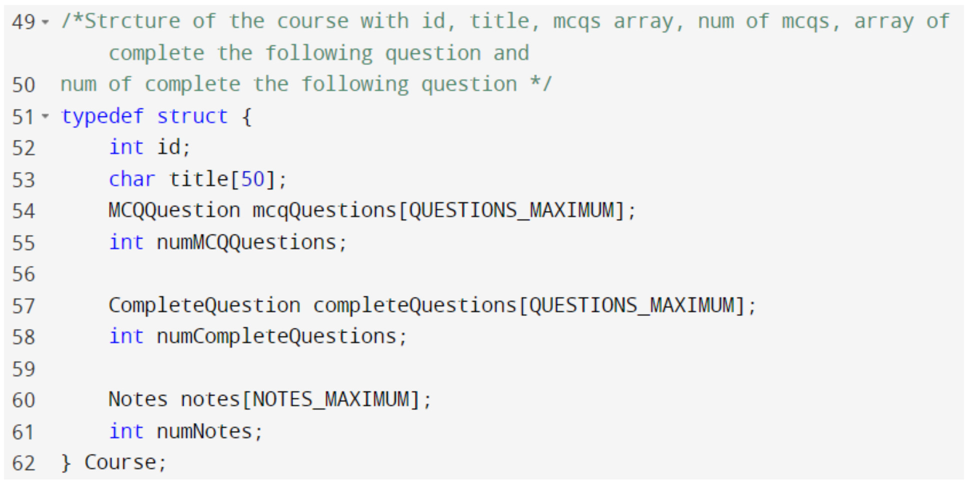
```c
/* Defining some structures for the entities of the program
Notice; C is not considered OOP but procedural, therefore we will use
    structures */

/* Structure of the MCQ with an id, title */
typedef struct {
    int id;
    char title[50];
} MCQQuestion;

/* Structure of complete the following question with an id, title */
typedef struct {
    int id;
    char title[50];
    char answer[100];
} CompleteQuestion;

/*Strcture of the admin user with id, name, and password */
typedef struct {
    int id;
    char name[50];
    char password[50];
} Admin;
```

Figure 21 Defining Some Structures for the Entities of The Program



```c
/*Strcture of the student user with id, name, and password, the courses that
    are enrolled, and number of courses enrolled */
typedef struct {
    int id;
    char name[50];
    char password[50];
    int enrolledCourses[COURSES_MAXIMUM];
    int numCourses;
} Student;
/*Strcture of the notes intered by user with id, array , name*/
typedef struct {
    int id;
    char content[200];
} Notes;
```

Figure 22 Defining Some Structures for the Entities of The Program



```c
/*Strcture of the course with id, title, mcqs array, num of mcqs, array of
    complete the following question and
num of complete the following question */
typedef struct {
    int id;
    char title[50];
    MCQQuestion mcqQuestions[QUESTIONS_MAXIMUM];
    int numMCQQuestions;

    CompleteQuestion completeQuestions[QUESTIONS_MAXIMUM];
    int numCompleteQuestions;

    Notes notes[NOTES_MAXIMUM];
    int numNotes;
} Course;
```

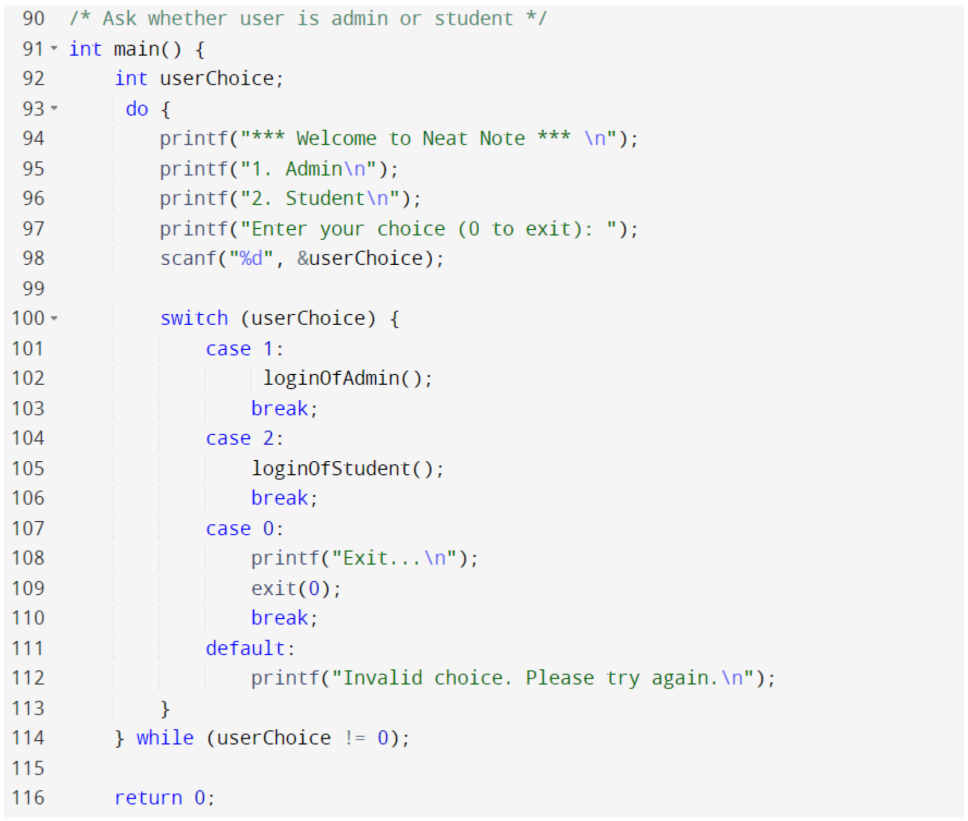
Figure 23 Defining Some Structures for the Entities of The Program

This system has many functionalities and features. In this section, all system's functionalities and features will be discussed and shown:

- **Welcome to the system.**

When the users enter the system, three options will be shown so the users can choose from. The option is passed to switch statement so the program can proceed to the next step based on the entry. The while loop is provided to let the user choose to access the system as Admin, student, Exit, or try again if the entry was wrong. As seen in the figures below:

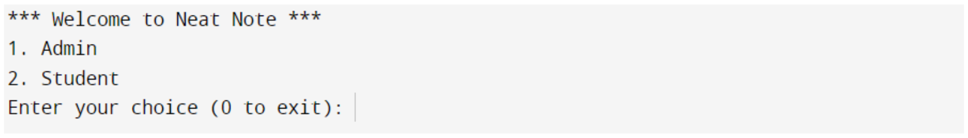
The code:



```c
/* Ask whether user is admin or student */
int main() {
    int userChoice;
     do {
        printf("*** Welcome to Neat Note *** \n");
        printf("1. Admin\n");
        printf("2. Student\n");
        printf("Enter your choice (0 to exit): ");
        scanf("%d", &userChoice);

        switch (userChoice) {
            case 1:
                 loginOfAdmin();
                break;
            case 2:
                loginOfStudent();
                break;
            case 0:
                printf("Exit...\n");
                exit(0);
                break;
            default:
                printf("Invalid choice. Please try again.\n");
        }
    } while (userChoice != 0);

    return 0;
```

Figure 24 Welcoming Code

The corresponding output:



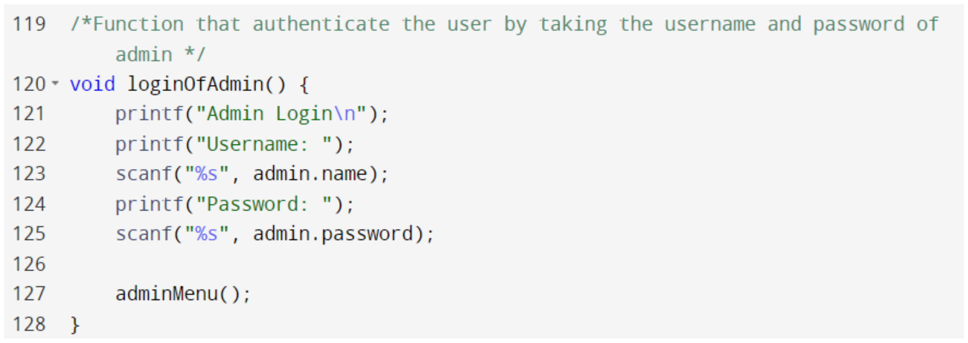
```
*** Welcome to Neat Note ***
1. Admin
2. Student
Enter your choice (0 to exit):
```

Figure 25 Welcoming Output

- **Admin’s section**

When the user is an Admin, they should type 1, and log in to the system, and then the admin’s functionalities will appear to allow the editor to choose one option at a time. The first functionality is user authentication using the function (loginOfAdmin()) to ask the users about their username and password. As seen in the following figures:

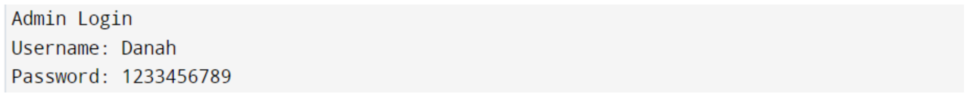
The code:



```c
/*Function that authenticate the user by taking the username and password of
    admin */
void loginOfAdmin() {
    printf("Admin Login\n");
    printf("Username: ");
    scanf("%s", admin.name);
    printf("Password: ");
    scanf("%s", admin.password);

    adminMenu();
}
```

Figure 26 Admin’s Section Code

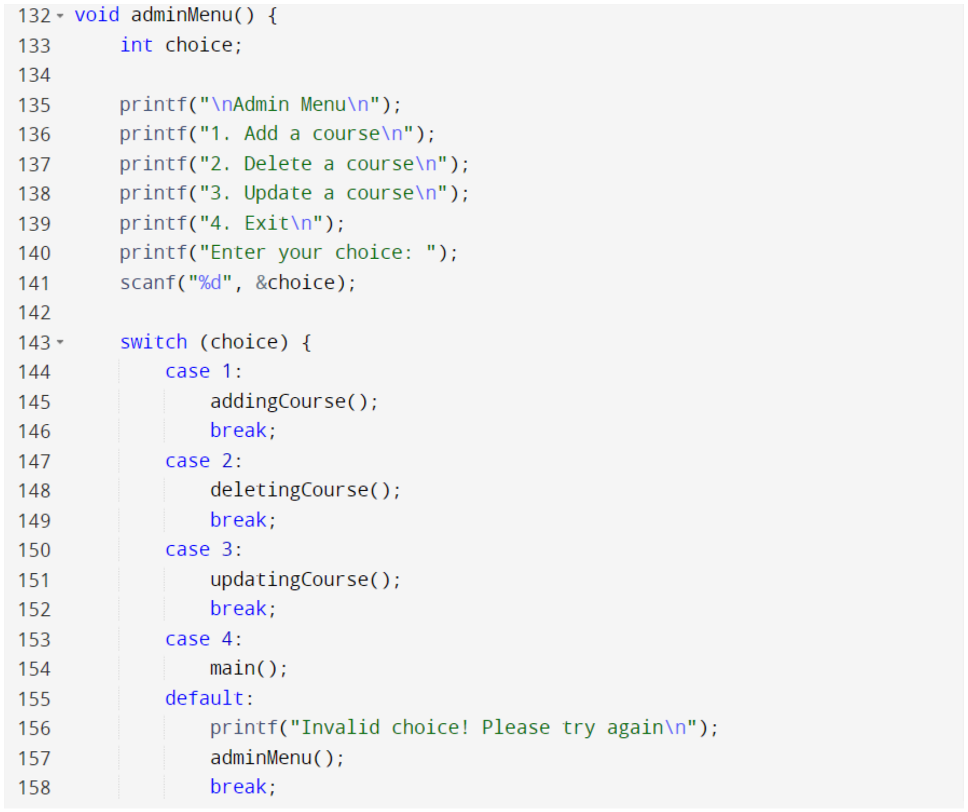
The corresponding output:



```
Admin Login
Username: Danah
Password: 1233456789
```

Figure 27 Admin’s Section Output

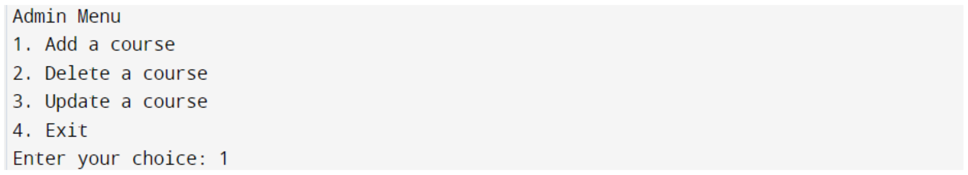
After the log in is done successfully the previous function will call (adminMenu()) function where the admin can choose to add, delete, update the course or exit to the main menu. One option at a time, the option is passed to switch statement so the program can proceed to the next step based on the entry.



```c
void adminMenu() {
    int choice;

    printf("\nAdmin Menu\n");
    printf("1. Add a course\n");
    printf("2. Delete a course\n");
    printf("3. Update a course\n");
    printf("4. Exit\n");
    printf("Enter your choice: ");
    scanf("%d", &choice);

    switch (choice) {
        case 1:
            addingCourse();
            break;
        case 2:
            deletingCourse();
            break;
        case 3:
            updatingCourse();
            break;
        case 4:
            main();
        default:
            printf("Invalid choice! Please try again\n");
            adminMenu();
            break;
```

Figure 28 Admin's Menu Code

The corresponding output:



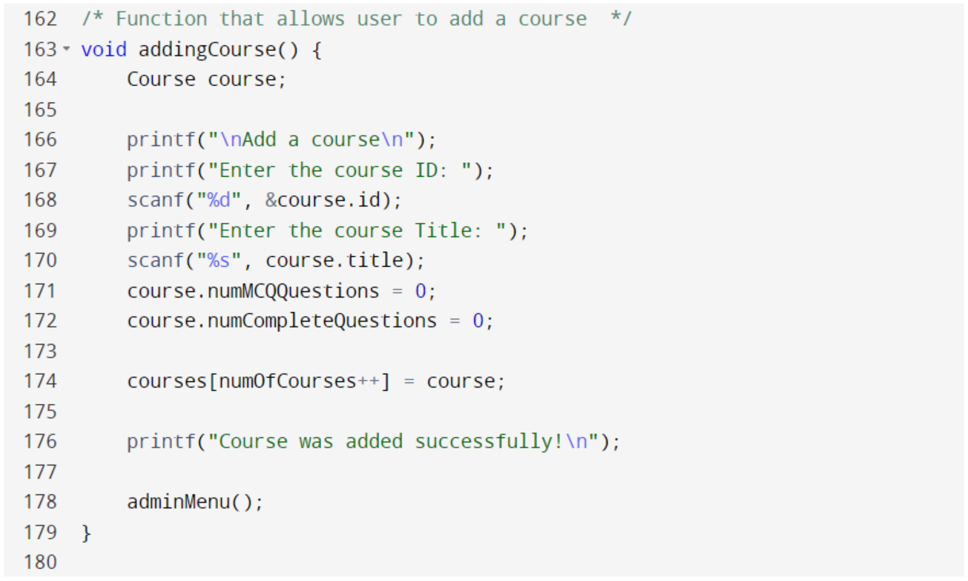
```
Admin Menu
1. Add a course
2. Delete a course
3. Update a course
4. Exit
Enter your choice: 1
```

Figure 29 Admin's Menu Output

- **Adding a Course**

Adding the course is done by entering course ID and course title. The entered information will be stored at the course structure where the array is initialized.

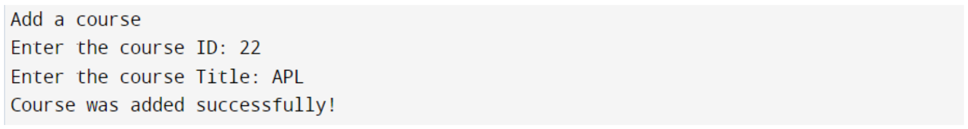
The code:



```c
/* Function that allows user to add a course  */
void addingCourse() {
    Course course;

    printf("\nAdd a course\n");
    printf("Enter the course ID: ");
    scanf("%d", &course.id);
    printf("Enter the course Title: ");
    scanf("%s", course.title);
    course.numMCQQuestions = 0;
    course.numCompleteQuestions = 0;

    courses[numOfCourses++] = course;

    printf("Course was added successfully!\n");

    adminMenu();
}

```

Figure 30 Adding Course Code

The corresponding output:



```
Add a course
Enter the course ID: 22
Enter the course Title: APL
Course was added successfully!
```

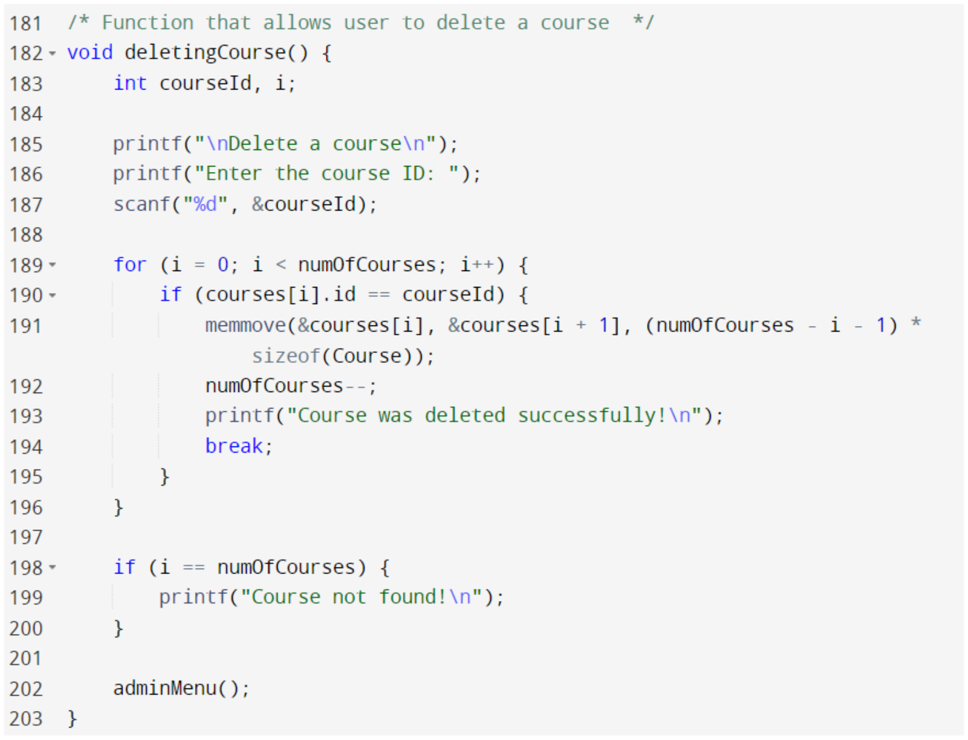
Figure 31 Adding Course Output

- **Deleting a Course**

Deleting the course is done by entering the course ID. The entered information will be processed using (memmove()) built in function also the array size will be decreased to make sure to delete the information after finding a match for the entered ID from the course structure where the array at. The function search for the course after the deletion to make sure it is deleted.

Note: memmove() function is used to overlap on the same memory or copy a certain amount of bytes from one memory to another.

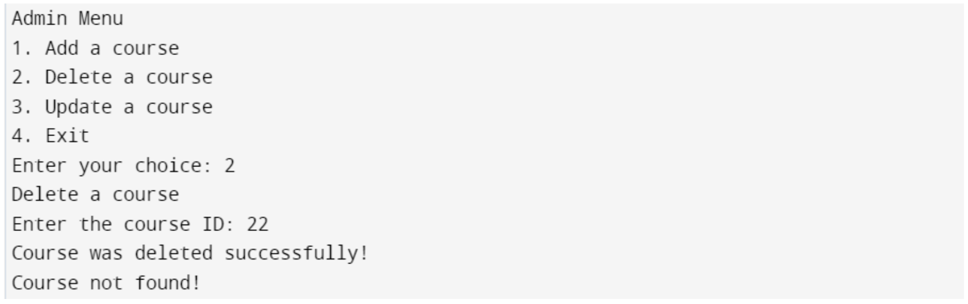
The code:



```c
/* Function that allows user to delete a course  */
void deletingCourse() {
    int courseId, i;

    printf("\nDelete a course\n");
    printf("Enter the course ID: ");
    scanf("%d", &courseId);

    for (i = 0; i < numOfCourses; i++) {
        if (courses[i].id == courseId) {
            memmove(&courses[i], &courses[i + 1], (numOfCourses - i - 1) *
                sizeof(Course));
            numOfCourses--;
            printf("Course was deleted successfully!\n");
            break;
        }
    }

    if (i == numOfCourses) {
        printf("Course not found!\n");
    }

    adminMenu();
}
```

Figure 32 Deleting Course Code

The corresponding output:



```
Admin Menu
1. Add a course
2. Delete a course
3. Update a course
4. Exit
Enter your choice: 2
Delete a course
Enter the course ID: 22
Course was deleted successfully!
Course not found!
```

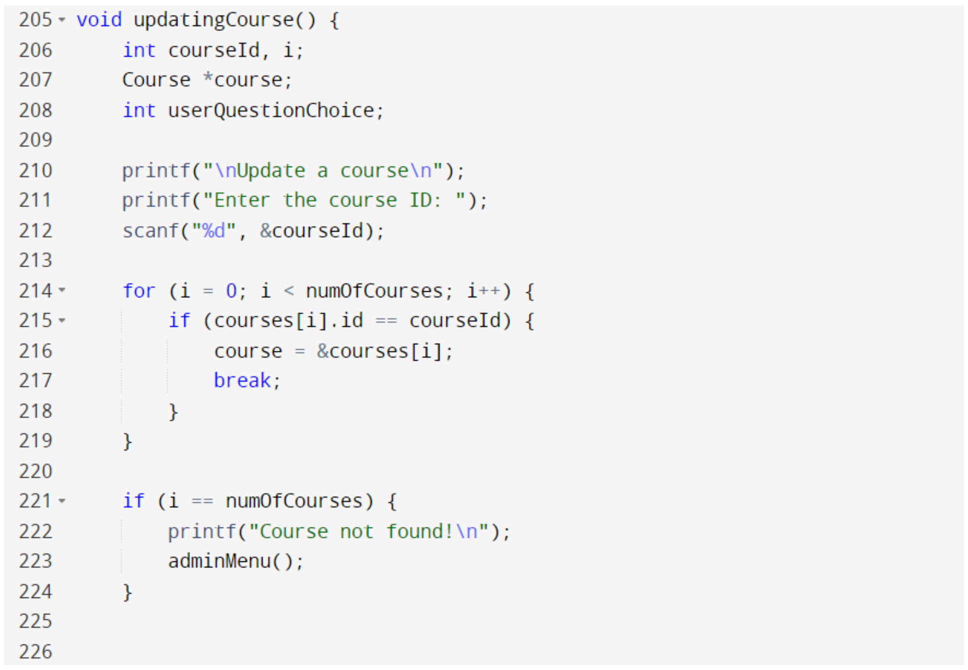
Figure 33 Deleting Course Output

- **Updating a Course**

Updating the course is done by entering type of question to update, the course ID, number of questions, and the new question. The entered information will be processed using (Scanf()) (fgets ()) (sscanf()) (getchar()) built in functions also the array will be accessed to switch the old content with new content or to add new content to array.

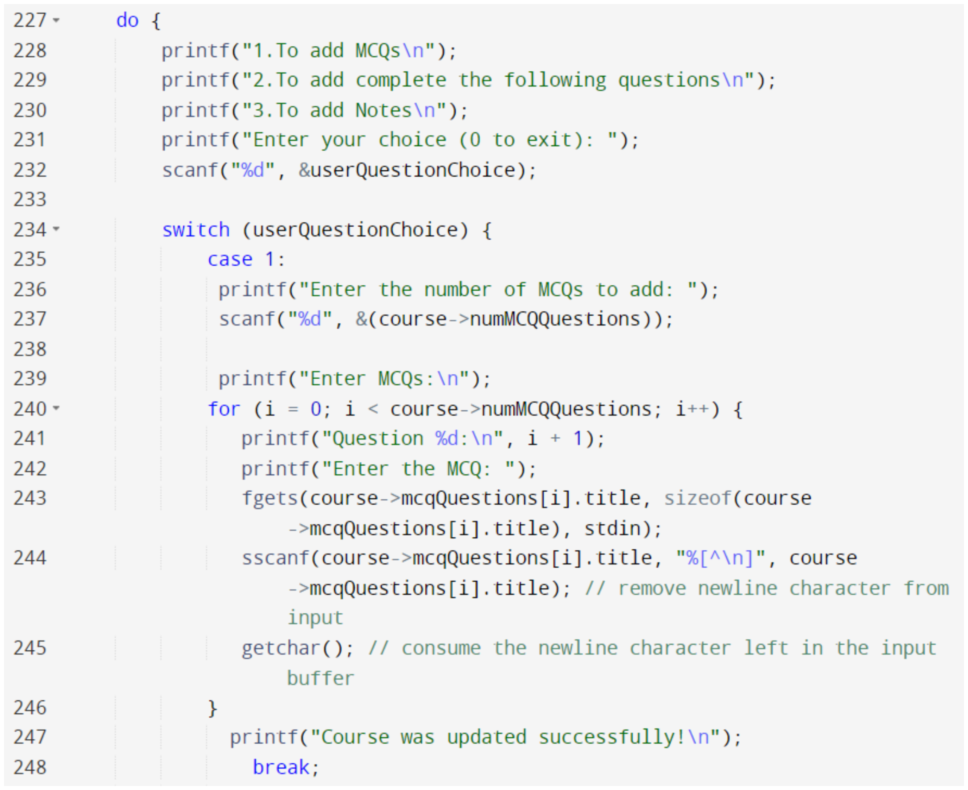
Note: Scanf() It enables you to read data from a file or the user and store that data in variables with various data kinds.

The code:



```c
void updatingCourse() {
    int courseId, i;
    Course *course;
    int userQuestionChoice;

    printf("\nUpdate a course\n");
    printf("Enter the course ID: ");
    scanf("%d", &courseId);

    for (i = 0; i < numOfCourses; i++) {
        if (courses[i].id == courseId) {
            course = &courses[i];
            break;
        }
    }

    if (i == numOfCourses) {
        printf("Course not found!\n");
        adminMenu();
    }


```

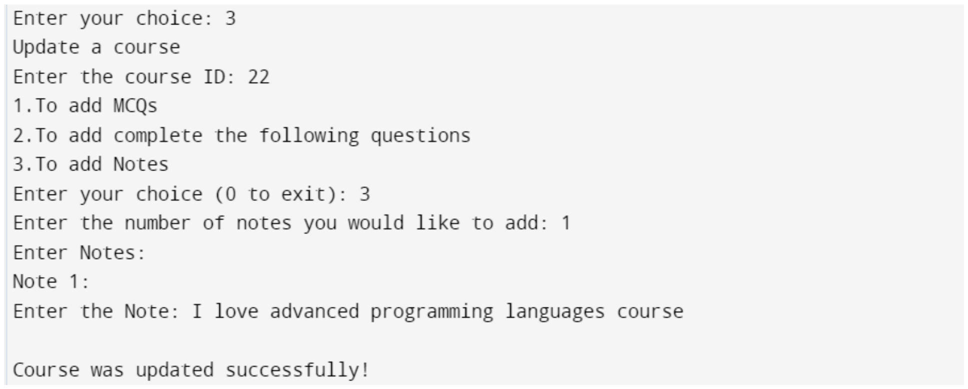
Figure 34 Updating Course Code



```
    do {
        printf("1.To add MCQs\n");
        printf("2.To add complete the following questions\n");
        printf("3.To add Notes\n");
        printf("Enter your choice (0 to exit): ");
        scanf("%d", &userQuestionChoice);

        switch (userQuestionChoice) {
            case 1:
             printf("Enter the number of MCQs to add: ");
             scanf("%d", &(course->numMCQQuestions));

             printf("Enter MCQs:\n");
            for (i = 0; i < course->numMCQQuestions; i++) {
                printf("Question %d:\n", i + 1);
                printf("Enter the MCQ: ");
                fgets(course->mcqQuestions[i].title, sizeof(course->mcqQuestions[i].title), stdin);
                sscanf(course->mcqQuestions[i].title, "%[^\n]", course->mcqQuestions[i].title); // remove newline character from input
                getchar(); // consume the newline character left in the input buffer
            }
              printf("Course was updated successfully!\n");
                break;
```

Figure 35 Updating Course Code

The corresponding output:



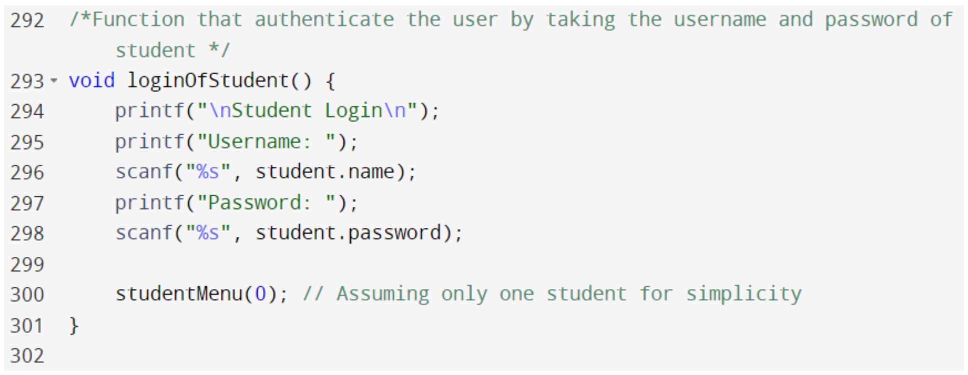
```
Enter your choice: 3
Update a course
Enter the course ID: 22
1.To add MCQs
2.To add complete the following questions
3.To add Notes
Enter your choice (0 to exit): 3
Enter the number of notes you would like to add: 1
Enter Notes:
Note 1:
Enter the Note: I love advanced programming languages course

Course was updated successfully!
```

Figure 36 Updating Course Output

- **Student's section**

When the user is an Student, they should type 2, and log in to the system, and then the Student's functionalities will appear to allow the Student to choose one option at a time. The first functionality is Student authentication using the function (loginOfStudents()) to ask the Student about their username and password. As seen in the following figures:

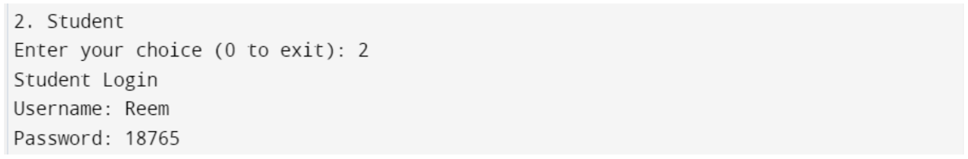
The code:



```c
/*Function that authenticate the user by taking the username and password of
    student */
void loginOfStudent() {
    printf("\nStudent Login\n");
    printf("Username: ");
    scanf("%s", student.name);
    printf("Password: ");
    scanf("%s", student.password);

    studentMenu(0); // Assuming only one student for simplicity
}
```

Figure 37 Student's Section Code

The corresponding output:

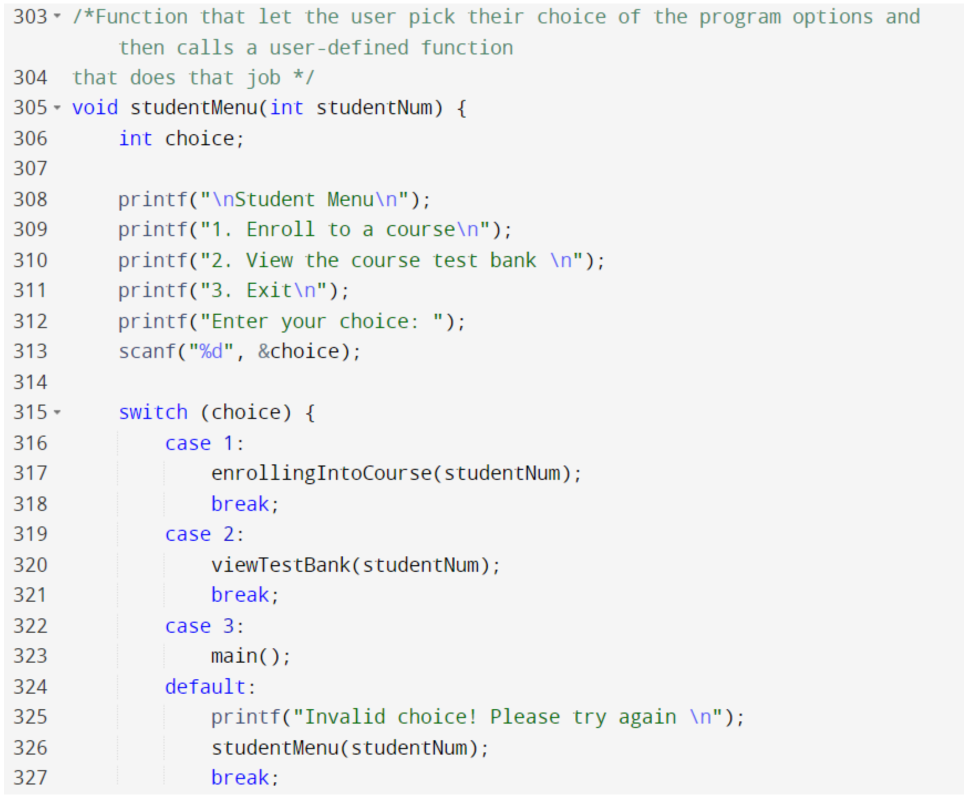


```
2. Student
Enter your choice (0 to exit): 2
Student Login
Username: Reem
Password: 18765
```

Figure 38 Student's Section Output

After the log in is done successfully the previous function will call (studentMenu()) function where the Student can choose to Enroll to a course, View the course test bank, or exit to the main menu. One option at a time, the option is passed to switch statement so the program can proceed to the next step based on the entry.

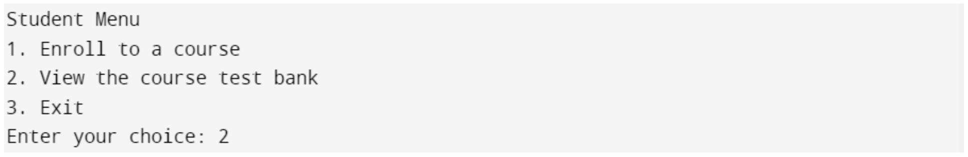
The code:



```c
/*Function that let the user pick their choice of the program options and then calls a user-defined function
that does that job */
void studentMenu(int studentNum) {
    int choice;

    printf("\nStudent Menu\n");
    printf("1. Enroll to a course\n");
    printf("2. View the course test bank \n");
    printf("3. Exit\n");
    printf("Enter your choice: ");
    scanf("%d", &choice);

    switch (choice) {
        case 1:
            enrollingIntoCourse(studentNum);
            break;
        case 2:
            viewTestBank(studentNum);
            break;
        case 3:
            main();
        default:
            printf("Invalid choice! Please try again \n");
            studentMenu(studentNum);
            break;
```

Figure 39 Student's Menu Code

The corresponding output:

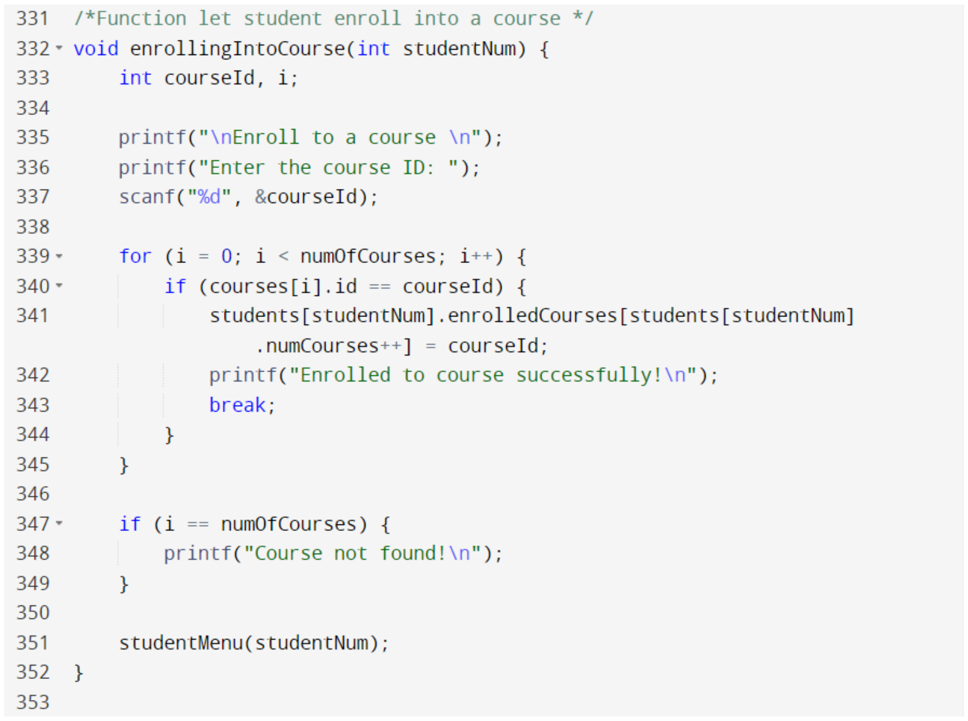


```
Student Menu
1. Enroll to a course
2. View the course test bank
3. Exit
Enter your choice: 2
```

Figure 40 Student's Menu Output

- **Enroll into a course.**

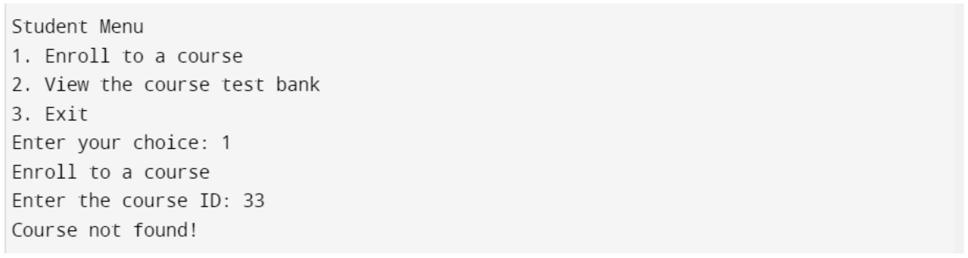
The code:



```c
/*Function let student enroll into a course */
void enrollingIntoCourse(int studentNum) {
    int courseId, i;

    printf("\nEnroll to a course \n");
    printf("Enter the course ID: ");
    scanf("%d", &courseId);

    for (i = 0; i < numOfCourses; i++) {
        if (courses[i].id == courseId) {
            students[studentNum].enrolledCourses[students[studentNum]
                .numCourses++] = courseId;
            printf("Enrolled to course successfully!\n");
            break;
        }
    }

    if (i == numOfCourses) {
        printf("Course not found!\n");
    }

    studentMenu(studentNum);
}

```

Figure 41 Enroll Into a Course Code

The corresponding output:



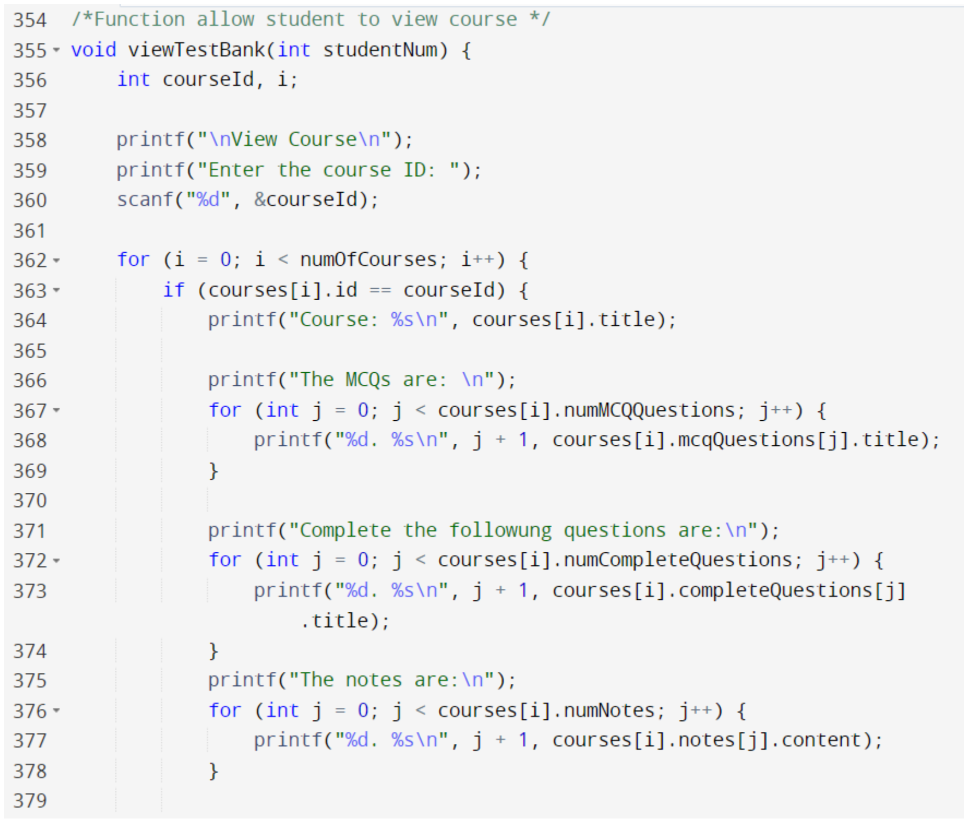
```
Student Menu
1. Enroll to a course
2. View the course test bank
3. Exit
Enter your choice: 1
Enroll to a course
Enter the course ID: 33
Course not found!
```

Figure 42 Enroll Into a Course Output

- **View a course test bank.**

This function prints all the entered questions by the Admin to the user based on the entered Course ID. It find a match using (Scanf()) built in function then prints all the data using loops.

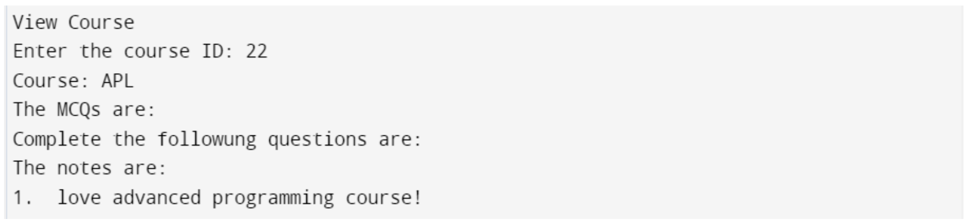
The code:



```c
/*Function allow student to view course */
void viewTestBank(int studentNum) {
    int courseId, i;

    printf("\nView Course\n");
    printf("Enter the course ID: ");
    scanf("%d", &courseId);

    for (i = 0; i < numOfCourses; i++) {
        if (courses[i].id == courseId) {
            printf("Course: %s\n", courses[i].title);

            printf("The MCQs are: \n");
            for (int j = 0; j < courses[i].numMCQQuestions; j++) {
                printf("%d. %s\n", j + 1, courses[i].mcqQuestions[j].title);
            }

            printf("Complete the followung questions are:\n");
            for (int j = 0; j < courses[i].numCompleteQuestions; j++) {
                printf("%d. %s\n", j + 1, courses[i].completeQuestions[j]
                    .title);
            }
            printf("The notes are:\n");
            for (int j = 0; j < courses[i].numNotes; j++) {
                printf("%d. %s\n", j + 1, courses[i].notes[j].content);
            }

```

Figure 43 View a Course Test Bank Code

The corresponding output:



```
View Course
Enter the course ID: 22
Course: APL
The MCQs are:
Complete the followung questions are:
The notes are:
1.  love advanced programming course!
```

Figure 44 View a Course Test Bank Output

9. Comparison between two programming languages

- **Part1:**

	C Programming Language	**Java Programming Language**
Language Paradigm	Procedural Programming Language, represents the codes as a sequence of statements [3].	Object-Oriented language, essentially relies on sequential processing with a changing amount of memory locations, focusing on data objects [3].
Syntax	C is a low to middle-level language that is closer to machine language than java, therefore it's more complex and not interpreted but only compiled [4].	Java is a high-level language whose syntax is easier to understand and uses a compiler or interpreter to translate code into machine language [4].
Dependencies	C is platform-dependent, it's supported by the concept of Write Once Compile Anywhere [4].	Java is platform-independent intended to have a minimal amount of implementation requirements, and support Write Once Read Anywhere, or WORA. Compiled Java code will execute on any platform that accepts it [4].
Memory Management	Memory allocation in C must be done explicitly by the functions 'calloc()' and 'malloc()', and for deallocation memory spaces function 'free()' is used, finally, garbage collecting must be done by the programmer manually [5].	Memory allocation and deallocation in java can be done automatically by a built-in garbage collector [5].

Table 1 Comparison between two programming languages based on (Language Paradigm, Syntax, Dependencies, Memory Management)

- **Part 2:**

Comparison 1: Login Function

Function	C Programming Language	Code Syntax

loginOfAdmin()	This function of type Void because there are no return values.
	The function 'printf()' will display a string on the user's screen.
	The code will ask the admin to enter the username and then the password, the 'scanf()' function will read the user's input without whitespaces and stores it inside variables 'admin.name' and 'admin.password'. After the admin successfully login to the system, 'adminMenu()' function will be called.
	C uses semi-colon at the end of each terminated statement.
		`void loginOfAdmin() {`
		`printf("Admin Login\n");`
		`printf("Username: ");`
		`scanf("%s", admin.name);`
		`printf("Password: ");`
		`scanf("%s", admin.password);`
		
		`adminMenu();`
		`}`

Table 2 Comparison between two programming languages based on Login Function (C)

Function	Java Programming Language	Code Syntax

login()	Since Java is an object-oriented language that supports inheritance, a Person class was created and the login() method was implemented in this class, and since the Editor class (Admin in C) extends Person Class, this method was used easily. This method of Boolean types returns true or false depending on the input, also it takes two different parameters username and password, and checks if the username and password that were	• **Login method in Person Class**
		`public boolean login(String username, String password) {`
		`return this.username.equals(username) && this.password.equals(password);`
		`}`
		• **Creating an object**
		`Editor editor = new Editor("Reem", "123");`
		• **Testing the inputs**
		`editor.login(username, password)`

	passed to the parameters are true or not, this happens by creating an object from Editor class and passing the username and password that were entered by the user to the constructor, then these username and password are then being passed to the parameters of this method to check the inputs.	

Table 3 Comparison between two programming languages based on Login Function (Java)

Comparison 2: Adding function

Function	**C Programming Language**	**Code Syntax**
addingCourse( )	This function will allow users to add a course, its type is Void because there are no return values. First the function will create a variable from a custom data type named ‘Course’, then scan the user’s input with ‘scanf()’ to store ‘course.id’ and ‘course.title’. The `&` symbol in front of `the variables are used to pass the memory address to `scanf()`. Then it sets the variables ‘course.numMCQQuestions’ and ‘course.numCompleteQuestions’ to zero initially. Then add it to the Course array, The `numOfCourses` variable is used to count the number of courses, and the postfix `++` operator increments `numOfCourses` after the new course has been successfully added.	void addingCourse() { Course course; printf("\nAdd a course\n"); printf("Enter the course ID: "); scanf("%d", &course.id); printf("Enter the course Title: "); scanf("%s", course.title); course.numMCQQuestions = 0; course.numCompleteQuestions = 0; courses[numOfCourses++] = course; printf("Course was added successfully!\n"); adminMenu(); }

		void addingCourse() {
		    Course course;

		printf("\nAdd a course\n");
		printf("Enter the course ID: ");
		scanf("%d", &course.id);
		printf("Enter the course Title: ");
		scanf("%s", course.title);
		course.numMCQQuestions = 0;
		
		course.numCompleteQuestions = 0;
		
		courses[numOfCourses]

Table 4 Comparison between two programming languages based on Adding course to the system (C)

Function	**Java Programming Language**	**Code Syntax**

Course()
AddNote()
	In Java code the course is created manually by creating an object from the Course class, the name of the course will be passed to the constructor of the class after the editor enters the name of it. After creating the course all methods inside the Course class can be used. Such as AddNote(), this method of type void takes one parameter “note” and has an ArrayList of type string in its body and uses add() built-in function with the “notes” object of the ArrayList to store the notes. After creating the object from the Course class, this method can be used and passed into it the note that the editor entered.	• **Creating an object from Course class**
		Course course = new Course(name);
		• **AddingNotes**
		public void addNote(String note) {
		notes.add(note);
		}
		• **Passing the note**
		course.addNote(note);

Table 5 Comparison between two programming languages based on Adding course to the system (Java)

Comparison 3: Deleting function

Function	**C Programming Language**	**Code Syntax**

deletingCourse()	This function will allow users to delete a course, its type is Void because there are no return values.
	The function will declare integer as ‘courseId’, and ‘I’ as a loop variable.
	The `scanf()` function scans the user's input and stores it in the `courseId` variable. The `&` symbol in front of `courseId` is used to pass the memory address of `courseId` to the function.
	The for loop will start from 0 and until it reaches the numOfCourses, the loop checks if the array contains the same id entered by the user, If the id is found, the \`memmove()\` function is used to move all of the elements after the matching id one position to the left in the array. This will delete the matching id course from the array. The \`numOfCourses\` variable is decreased by 1. Every question associated with the course will be deleted.
	If the ‘i’ reaches numOfCourses it means that course is not found.
		void deletingCourse() {
		    int courseId, i;
		
		    printf("\nDelete a course\n");
		    printf("Enter the course ID: ");
		    scanf("%d", &courseId);
		
		    for (i = 0; i < numOfCourses; i++) {
		        if (courses[i].id == courseId) {
		            memmove(&courses[i], &courses[i + 1], (numOfCourses - i - 1) * sizeof(Course));
		            numOfCourses--;
		            printf("Course was deleted successfully!\n");
		            break;
		        }
		    }
		
		    if (i == numOfCourses) {
		        printf("Course not found!\n");
		    }
		
		    adminMenu();
		}

Table 6 Comparison between two programming languages based on deleting course from the system (C)

Function	**Java Programming Language**	**Code Syntax**

deleteNote()	Since the course was created manually, it cannot be deleted by the editor, but the notes and other questions can be deleted. This method in the Course class of type void has one parameter which is the note to be deleted (the last added question). The body of this method contains an ArrayList of string type and uses the "notes" object to delete the note by calling the built-in function remove() that removes the last added question. Then by the course object this method can be used and passed the note to be deleted in the main class	• **Deleteing a note**
		public void deleteNote(String note) {
		      notes.remove(note);
		  }
		• **Passing the note**
		course.deleteNote(note);

Table 7 Comparison between two programming languages based on deleting course from the system (Java)

Comparison 4: displaying function

Function	**C Programming Language**	**Code Syntax**

	This function will allow users to delete a course, its type is Void because there are no return values.
	The function begins by declaring two variables, an integer `courseId`, which will store the ID of the course to be displayed, and an integer `i`, which will be a loop counter.
	It has `for` loop to iterate through the courses in the `courses` array. The loop starts from index 0 to the value of `numOfCourses`.
	“%d” is used to read integer input from the user.
	After searching the Courses array for the entered id, the first `for` loop prints the multiple-choice questions associated with the course, the second `for` loop prints the complete the following questions associated with the course. the third `for` loop prints the notes associated with the course, “%s” is used as identifier for string inputs/outputs.
	If the ‘i’ reaches numOfCourses it means that course is not found.
		void viewTestBank(int studentNum)
		{
		    int courseId, i;
		
		    printf("\nView Course\n");
		    printf("Enter the course ID: ");
		    scanf("%d", &courseId);
		
		    for (i = 0; i < numOfCourses; i++)
		{
		        if (courses[i].id == courseId) {
		            printf("Course: %s\n",
		courses[i].title);
		
		            printf("The MCQs are: \n");
		            for (int j = 0; j <
		courses[i].numMCQQuestions; j++)
		{
		                printf("%d. %s\n", j + 1,
		courses[i].mcqQuestions[j].title);
		            }
		
		            printf("Complete the
		followung questions are:\n");
		            for (int j = 0; j <
		courses[i].numCompleteQuestions;
		j++) {
		                printf("%d. %s\n", j + 1,
		courses[i].completeQuestions[j].title);
		            }
		            printf("The notes are:\n");
		            for (int j = 0; j <
		courses[i].numNotes; j++) {
		                printf("%d. %s\n", j + 1,
		courses[i].notes[j].content);
		            }
		
		        }
		    }

		if (i == 0) {
		    printf("Course not found!\n");
		  }
		
		  studentMenu(studentNum);
		}

Table 8 Comparison between two programming languages based on displaying course from the system (c)

Function	**Java Programming Language**	**Code Syntax**

displayQuestions()	This method of void type is in the Course class and was created to display questions and notes by using the for each loop. This type of loop is suitable with ArrayLists since the variable String question and String note refers to each string (whether it is a note or a question) in the ArrayList and by System.out.println(question), and System.out.println(note); the question and the note will be displayed at each iteration. In the main class, this method will be called by using the course object.	• **Displaying questions and notes method**
		
		public void displayQuestions() {
		
		System.out.println("MCQ Questions:");
		    for (String question : mcq) {
		
		System.out.println(question);
		    }
		
		System.out.println("Complete Questions:");
		    for (String question : complete) {
		
		System.out.println(question);
		    }
		
		System.out.println("Notes:");
		    for (String note : notes) {
		
		System.out.println(note);
		    }
		  }
		
		• **Using the method in main class**
		course.displayQuestions();

Table 9 Comparison between two programming languages based on displaying course from the system (java)

10. Attachments:

Codes are attached as NotePad files.

11. References:

[1] "Java Programming Language." *Java Programming Language - an Overview | ScienceDirect Topics*, www.sciencedirect.com/topics/computer-science/java-programming-language. Accessed 6 June 2023.

[2] *The Bell System Technical Journal*. American Telephone and Telegraph Co., 1979.

[3] Cormen, Thomas H. Introduction to Algorithms. MIT Press, Cambridge, Mass, 2009.

[4] "Difference between Java and C Language." *GeeksforGeeks*, 21 Feb. 2023, www.geeksforgeeks.org/difference-between-java-and-c-language/.

[5] "Learn What Is the Difference between C and Java." *testbook.com*, testbook.com/key-differences/difference-between-c-and-java. Accessed 6 June 2023.