Adapting software engineering course and students' assessment to fully remote mode

Katerina Zdravkova, Dejan Gjorgjevikj,
Gjorgji Madjarov, Katarina Trojachanec Dineva
UKIM, Faculty of Computer Science and Engineering, Skopje
{katerina.zdravkova, dejan.gjorgjevikj, gjorgji.madjarov, katarina.trojacanec} @finki.ukim.mk

Overview

- Current curricula at FCSE
- Course syllabus
- Beginning of the course
- Shift of the course towards a fully remote mode
- Organizing reliable remote assessment
- Modification of the exam questions
- Preventing cheating
- Return to new normal
- Software engineering course in the new curricula

Current curricula at FCSE

- Accreditation period: 1 October 2018 – 30 September 2023
- Study programs:
  - Software engineering and information systems (in Macedonian and in English)
  - Internet, networks and security (in Macedonian only)
  - Application of information technologies (in Macedonian only)
  - Computer Education (in Macedonian only)
  - Computer Engineering (in Macedonian only)
  - Computer Science (in Macedonian only)
- Two possible degrees:
  - 3-year studies, 180 ECTS, Bachelor of information technologies
  - 4-year studies, 240 ECTS, Bachelor of computer science and engineering

Status of software engineering course

- 6 ECTS course
- Fourth semester
- Prerequisite: Object-oriented programming
- Compulsory for the following study programs:
  - Application of information technologies
  - Computer Science
- Elective for the following study programs:
  - Internet, networks and security
  - Computer Education
  - Computer Engineering

Course syllabus (theory)

First midterm

01. Introduction
02. Software quality
03. Software process models
04. Agile methodologies
05. Requirements engineering
06. System modeling

Second midterm

07. Architectural design
08. Component-level design
09. User interface design
10. Design and implementation
11. Software testing strategies
12. Software testing techniques
13. Software Evolution

Course syllabus (exercises)

First midterm

01. Version control systems
02. Introduction to Git
03. Git 1 (Git real, levels 1-3)
04. Git 2 (Git real, levels 4-6)
05. Git 3 (Git real 2)
06. Git workflows

Second midterm

07. UML Use Case
08. UML Activity Diagram
09. UML Class and Sequence Diagram
10. Build automation tools: Gradle
11. Testing
12. Unit testing – JUnit 5

Software engineering course before 2019

- Lectures/exercises
  - Semi-formal lectures on site, slide presentations (lecture room)
  - Exercises organized as demonstration lectures and problem-solving exercises (lecture room / computer laboratory)
  - Moodle as LMS for lecture/exercises materials distribution
- Assessment
  - Combination of multiple choice, short answer and essay questions
  - Moodle quizzes in controlled environment (computer laboratory, no internet access, no phones during the exam)

First release of the course in 2019/22

- After 3 weeks of lectures, COVID-19 pandemic was announced
- The course was immediately transformed into a remote mode
- Learning environment: BigBlueButton (Virtual Classroom Software)
- The lectures were pre-recorded
- Interactive Q&A sessions using BBB
- Study pack: chapters by Ian Sommerville and Roger Pressman

Video lectures and supporting study pack

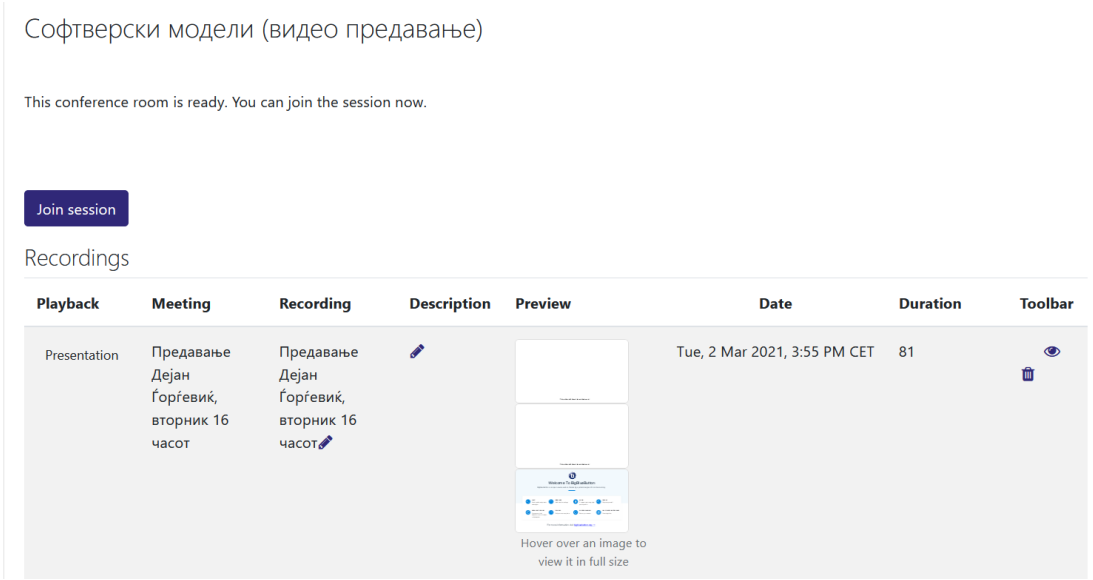
19 April - 25 April

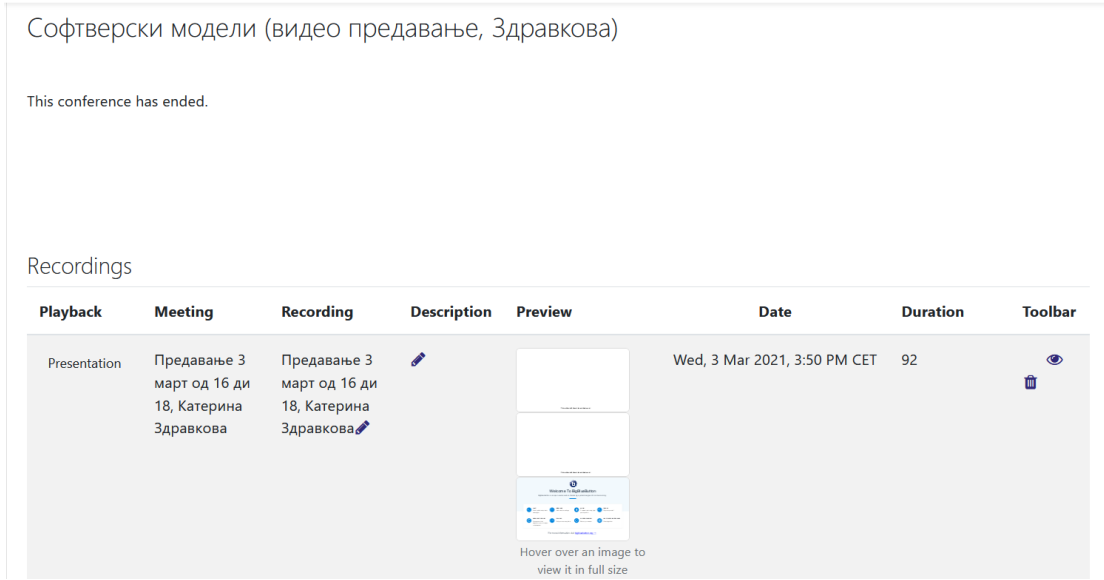
- Дизајн на кориснички интерфејси (предавање)
  - 09. User Interface Design
- Дизајн и имплементација (видео предавање)
  - 10. Design&Implementation
  - 10. Design&Implementation(Sommerville)
- UML Class and Sequence Diagram (аудиториска вежба 8)
- UML Class and Sequence Diagram - аудиториска вежба 8 (видео снимка)

26 April - 2 May

- Стратегии за тестирање (видео предавање)
  - 11. TestingStrategies
  - Software Testing Strategies (Pressman)
- Gradle (аудиториска вежба 9)
- Gradle - Аудиториска вежба бр.9 (видео-снимка)

Recorded lectures (one or two versions)





Students' assessment

Year	Enrolled students on the software engineering course
2022	570
2021	520
2020	680
2019	680

- Assessment environment:
  - BBB for communication
  - Moodle quiz for the exam
- Anti-cheating technologies:
  - Preliminary phase (30%, qualifying for the next 70%):
    - Safe Exam Browser
  - Advanced phases (40% essentials + 30% more complex questions)
    - Students share their screen and their working environment
    - Their microphone is turned on
    - Sharing of working environment is enabled by ManyCam or DroidCam
    - The exams are recorded to prove the prospective accusation of intentional cheating

Manuals for using ManyCam & DroidCam

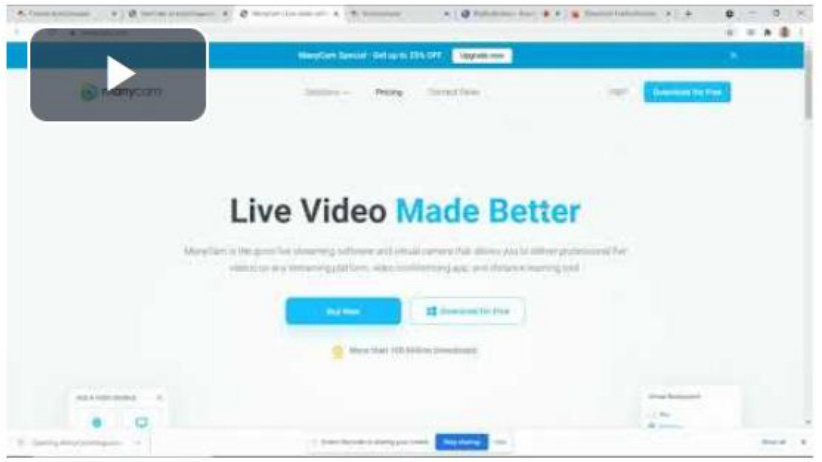
Упатства

Hidden from students

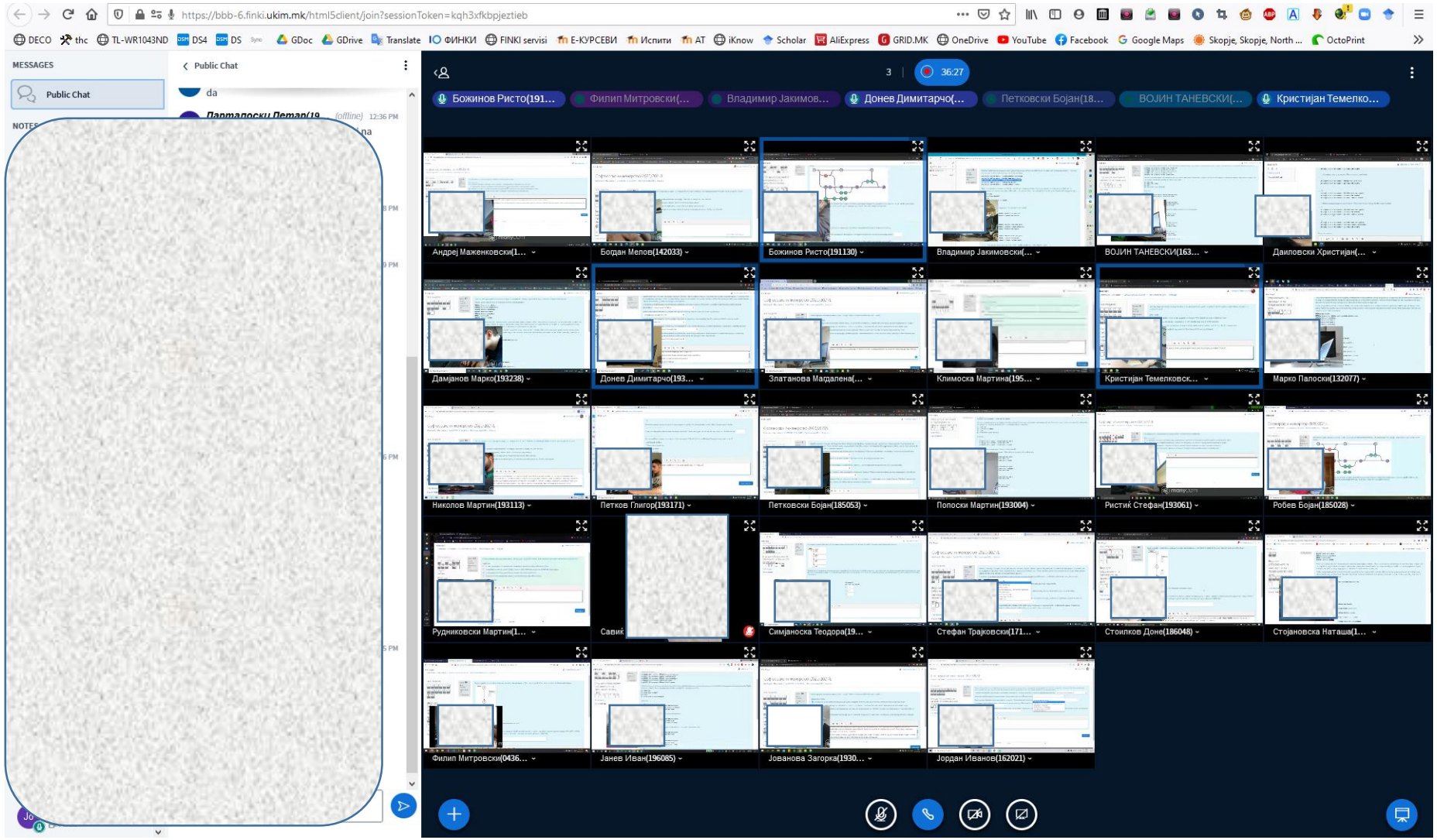
Упатство за користење на ManyCam

Упатство за користење на DroidCam

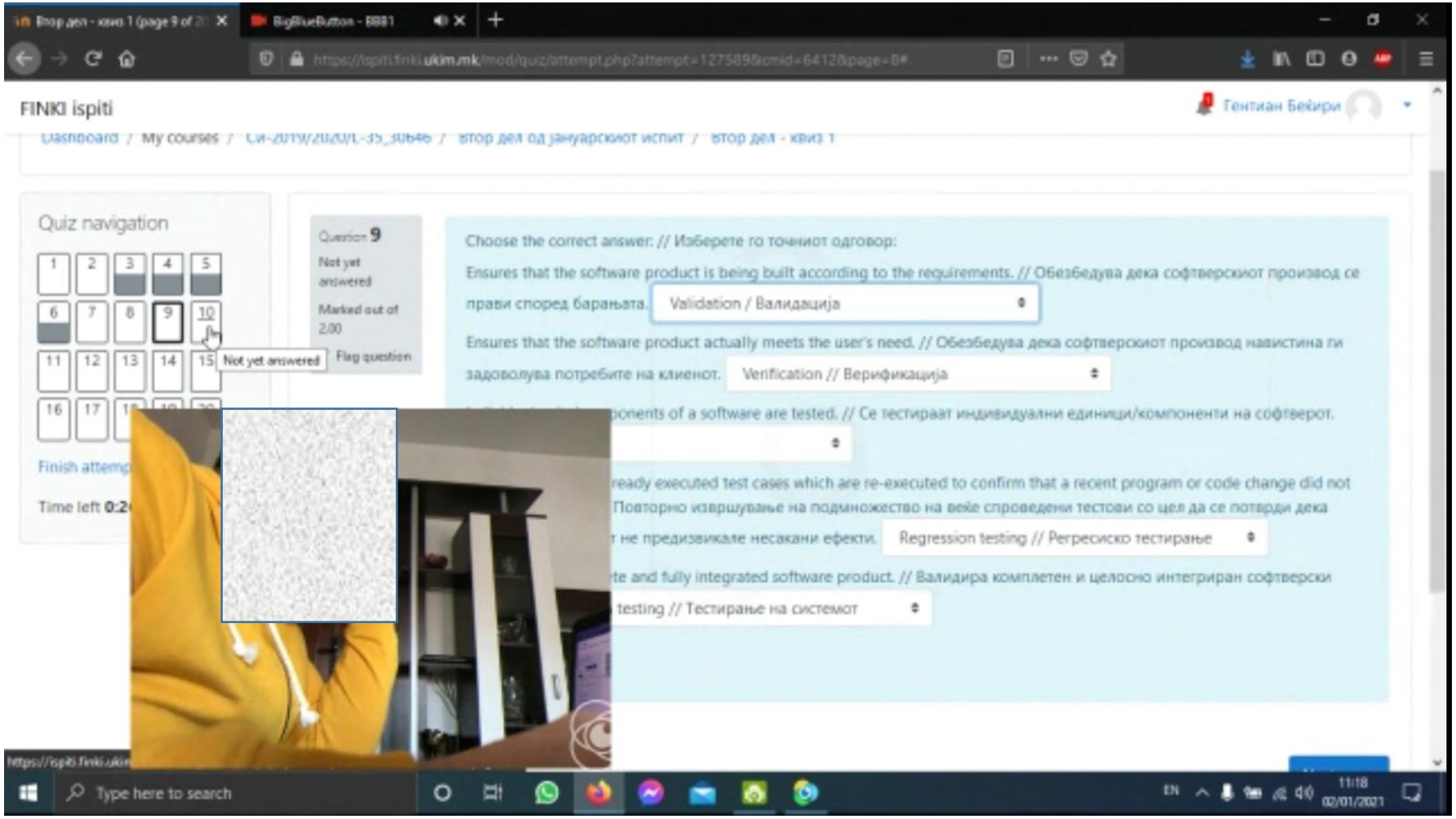
Упатство за далечинско полагање на испитите на ФИНКИ со апликацијата ManyCam



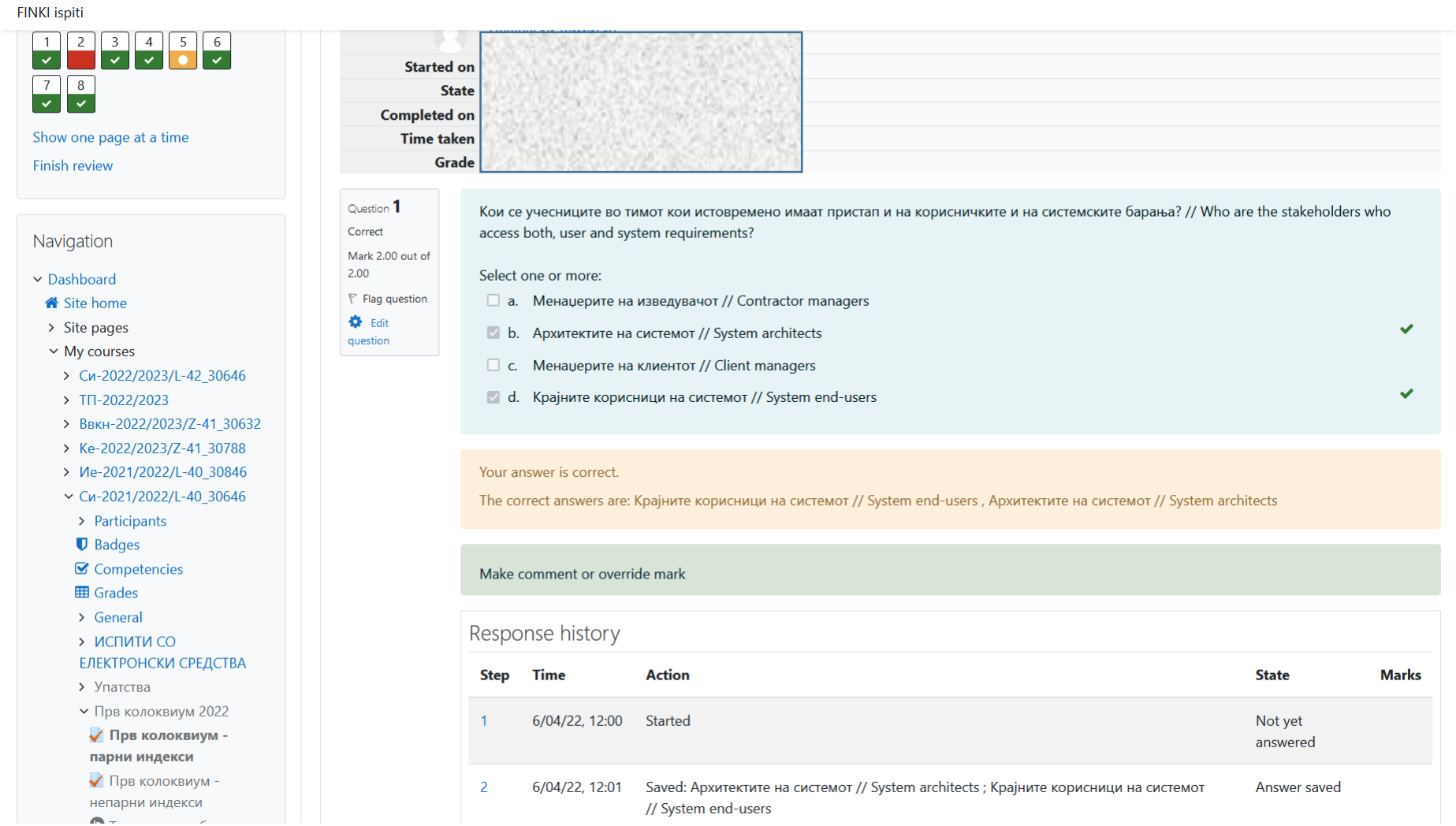
Monitoring of one classroom



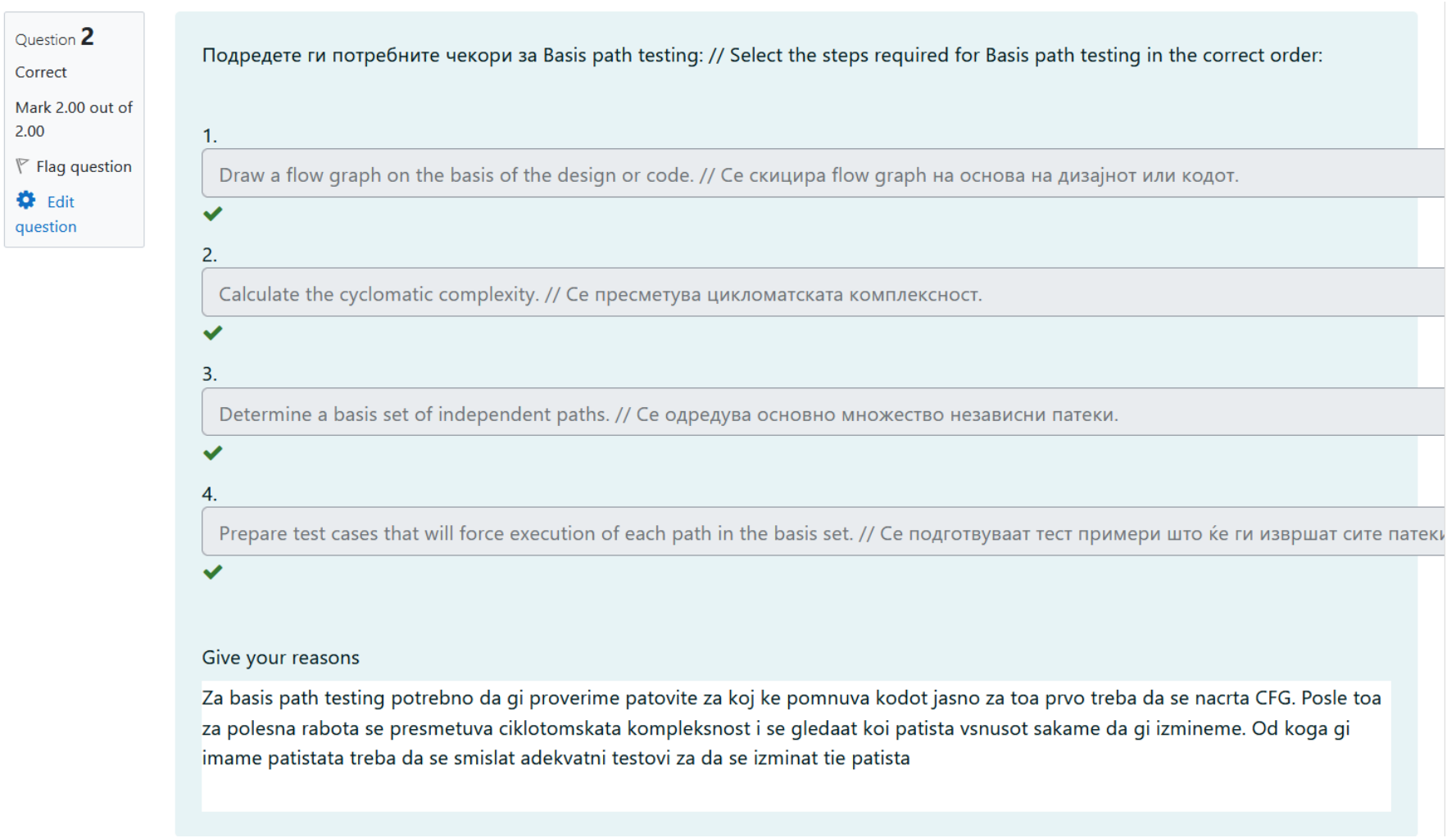
Monitoring of one student



Inside the exam (theoretical question)



Questions in the advanced phases need an explication



Inside the exam (practical question)

FINKI ispiti

Question **4**

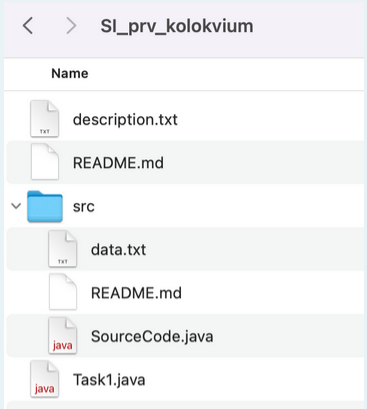
Correct

Mark 4.00 out of 4.00

Flag question

Edit question

Дадена е папката SI_prv_kolokvium (вооедно и локален Git репозиториум) со следната содржина:



Папката SI_prv_kolokvium содржи и друга папка именувана src.

Соодветно потоа се извршуваат следните чекори: // After that, the following steps are applyied:

1. Се додава код во датотеките `Task1.java` и `src/SourceCode.java`
2. Се додава текст во `README.md` и `src/README.md`
3. Се извршуваат командите `git add "*.java"` и `git add *.md`
4. Се извршува командата `git commit`. // The command `git commit` is run.
5. Се менува содржината на датотеките `description.txt` и `src/data.txt`.
6. Се менува содржината на датотеката `src/SourceCode.java`.

Во која фаза од животниот циклус на Git се наоѓа секоја од датотеките? // Select the git lifecycle phase for each of the files.

- `desciption.txt` untracked ✔
- `README.md` unmodified ✔
- `Task1.java` unmodified ✔
- `src/data.txt` untracked ✔
- `src/README.md` untracked ✔
- `src/SourceCode.java` modified ✔

Practical questions in the advanced phases need an explication too

Question **7**

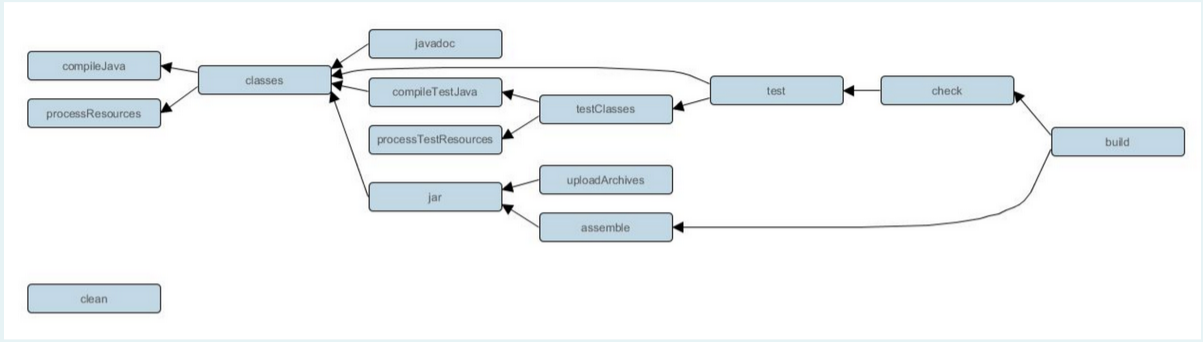
Correct

Mark 4.00 out of 4.00

Flag question

Edit question

Доколку развивате Java апликација со Gradle и не се вчитаат успешно некои од потребните ресурси за вашите тестови, кои таскови од Java plugin-от нема да се извршат? // If you are developing a Java application with Gradle build automation tool and some of the necessary resources for your tests are not loaded successfully, which of the tasks in the Java plugin will fail?



Select one or more:

- [ ] a. compileJava
- [ ] b. processResources
- [ ] c. classes
- [ ] d. javadoc
- [ ] e. compileTestJava
- [x] f. processTestResources ✔
- [ ] g. jar
- [x] h. testClasses ✔
- [ ] i. uploadArchives
- [ ] j. assemble
- [x] k. test ✔
- [x] l. check ✔
- [x] m. build ✔

Return to new normal

- The course was enrolled by 540 students divided into 5 study groups
- We decided to organize face-to-face “thematic lecturing”
- The introductory lecture is presented by all teachers to the study group they are responsible for
- In the following weeks, one teacher presents the same lecture to all the study groups
- Exhausting for the teacher
- All students get the same information (more or less) about the same theme

Face-to-face assessment

- The browser accesses only the exam site, disabling searching for the answer on the course site or on the Web
- The exam site cannot be accessed from an IP different from laboratory IP addresses
- Cell phones are collected and kept until students leave the lab
- The exam is integrated, with no qualifying and advanced phases
- The division is the following:
  - Questions with explanation
  - Theoretical questions
  - Practical questions

The exam in June 2023

Теорија (прашања со избор и објаснување)

1 2 3 4

Теорија

5 6 7 8 9

10 11 12 13 14

15 16 17 18

Практично

19 20 21 22 23

24 25 26

Show one page at a time

Finish review

Navigation

Started on	Tuesday, 27 June 2023, 10:00 AM
State	Finished
Completed on	Tuesday, 27 June 2023, 11:30 AM
Time taken	1 hour 30 mins
Grade	**77.32** out of 100.00

Question **1**

Partially correct

Mark 0.50 out of 3.00

Flag question

Edit question

Кои од следните практики **не се** користат во екстремно програмирање // Which of the following practices are not used in extreme programming.

Select one or more:

- [x] a. непрекината интеграција // continuous integration ✗
- [ ] b. клиентот е присутен во компанијата за време на развојот // on-site customer during the development
- [ ] c. инкрементално планирање // incremental planning
- [x] d. детален дизајн // detailed design ✓
- [ ] e. тестирање после кодирање // testing after coding

Give your reasons

Екстремното програмирање е техника која што се користи каде што вработените во компанијата имаат брзи спринтови односно работат во екстремно брзи услови и во мали групи за да добијат брзо решение на некоја предвидена задача. Интеграцијата секако е прекината со тоа што после некоја задача мора истата да с епровери, тестира и да се споредат резултатите. Некоја огромна поддршка од клиентот при овој тип на програмирање не очекуваме.

Детален дизајн секако не сакаме бидејќи целта ни е да гледаме голема слик

Practical question (part 1)

Question **24**

Correct

Mark 8.00 out of 8.00

Flag question

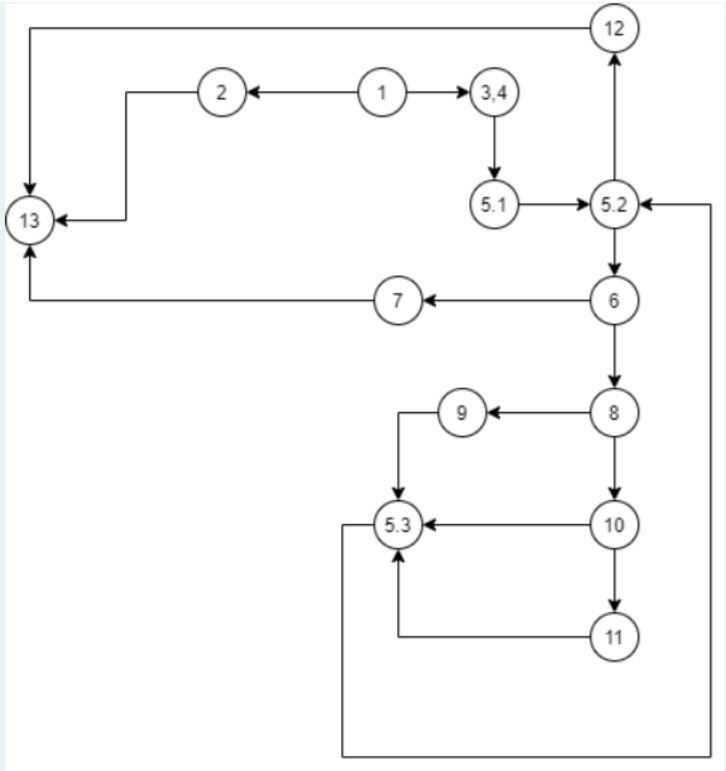
Edit question

Дадена е класата „JunePart3“ со методот „public static int getHomeAndAwayDifference(int[][] scores)“, каде врз основа на внесените резултати од фудбалски натпревари, се враќа разликата помеѓу бројот на победи на домашниот и гостинскиот тим. За истиот метод е даден и Control Flow Graph.

/

There is a class "JunePart3" with a method "public static int getHomeAndAwayDifference(int[][] scores)", which returns the difference between the number of wins of home and away teams based on the entered football scores. For the same method, there is a Control Flow Graph.

```
public class JunePart3 {
    public static int getHomeAndAwayDifference(int[][] scores){
        if (scores == null || scores.length <= 0){                          // 1
            throw new RuntimeException("Have to pass at least one score!"); // 2
        }
        int homeWins = 0;                                                   // 3
        int awayWins = 0;                                                   // 4
        for (int i = 0; i < scores.length; i++){                            // 5
            if (scores[i].length != 2){                                     // 6
                throw new RuntimeException("All score arrays must have a length of 2!"); // 7
            } if (scores[i][0] > scores[i][1]){                             // 8
                homeWins += 1;                                              // 9
            } else if (scores[i][0] < scores[i][1]){                        // 10
                awayWins += 1;                                              // 11
            }
        }
        return homeWins-awayWins;                                           // 12
    }                                                                       // 13
}
```



Одговорете на следните прашања: / Answer the following questions:

Кој е минималниот број на тестови кои треба да се напишат за да се исполни условот за:

/ What is the minimum number of test cases that need to be written to meet the requirement for:

- C0 - Every statement методата / C0 - Every statement method 3 ✔
- C1 - Every branch методата / C1 - Every branch method 3 ✔

Која е цикломатската комплексност? / What is the cyclomatic complexity? 6 ✔

A question that lacks an English translation ☺

Question **23**

Correct

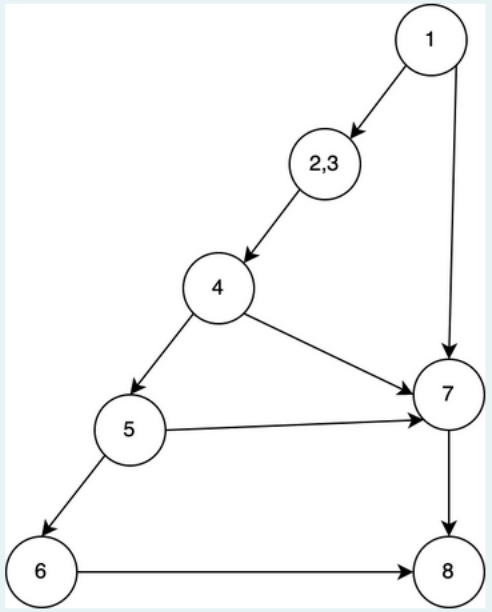
Mark 4.00 out of 4.00

Flag question

Edit question

Дадена е класата „JuneSI“ со методот „public boolean valid(String file)“, каде врз основа на името на фајлот се враќа вредност дали неговото име е валидно. За истиот метод е даден и Control Flow Graph.

```
public class JuneSI {
    public boolean valid(String file) {
        if (file.endsWith(".java")) { //1
            String[] parts = file.split( regex: "."); //2
            String fileName = parts[0]; //3
            if (fileName.length() == 4) { //4
                if (!fileName.contains("_") && !fileName.contains("-")) { //5
                    return true; //6
                }
            }
        }
        return false; //7
    } //8
}
```



Цикломатската комплексност изнесува: 4 ✔

Кој е минималниот број на тестови кои треба да се напишат за да се исполни условот за C0 Every statament? 2 ✔

Која патека се изминува со тестот: влез = "index.java" излез = false? 1-2-3-4-7-8 ✔

Software engineering course in the new accreditation

- Accreditation period: 1 October 2023 – 30 September 2028
- All the academic study programs and the status of the course remain unchanged
- New professional 2-year and 3-year study programs start attracting students
- Software engineering course is compulsory for 3-year studies and elective in the 2-year studies

Thank you for your attention

Questions ???