|  |  |  |  |
| --- | --- | --- | --- |
| 과목명 | Java 네트워크 프로그래밍 | 시행일자 | 2022.04.06 (수행시간 2~4시간) |
| 점수 |  | 제출자 | 강현규 |
| 아래 요구 사항에 맞는 제품을 제작하여 제출하시오   |  | | --- | | [요구사항]  성적관리 프로그램을 제작 의뢰합니다(ver 0.6.0)  각과목은국어, 영어, 수학로 3과목입니다  학생수는 20~30내외 입니다  학생의 정보는 학번과 이름, 각 과목의 성적, 평균점수(소수점 두자리)를 출력해주세요  수정과 삭제가 가능하도록 기능해야 합니다  사용자가 편리한 인터페이스 지원을 요구합니다  단, 콘솔 실행입니다  프로그램은 종료후 재시작 후에도 계속 데이터 유지되어야 합니다  오토세이브 기능으로 20초 단위로 자동저장되어 관리됩니다  학번은 최초 부여받고 변동 없습니다 학번서식은 1번부터 순서대로 부여됩니다 |  |  |  |  |  | | --- | --- | --- | --- | | 배점표 | | | | | 1 | 일정시한 엄수 | 20 |  | | 2 | 사용자 편의성(UI) | 5 |  | | 3 | 자료구조의 선택 | 5 |  | | 4 | 코드 가독성 | 5 |  | | 5 | 리펙토링 | 5 |  | | 6 | 문법 활용 능력 | 5 |  | | 7 | 제약조건 | 5 |  | | 8 | IO 제어능력 | 20 |  | | 9 | 스레드 제어능력 | 20 |  | | 10 | 문서화 | 5 |  | | 11 | 기타 | 5 |  | | 계 | | 100 |  | | | | |
| 제품어필 | | | |
| 1번을 입력하면 기존 데이터에 추가로 입력해서 파일을 저장합니다.  2번을 입력하면 기존 데이터를 삭제한 후 새로운 내용을 입력하여 파일로 저장합니다  3번을 입력하면 현재까지의 데이터를 출력하여 보여줍니다.  5번을 입력하면 지금까지의 데이터를 전체 삭제합니다. | | | |
| 출력결과 | | | |
| <스크린샷> | | | |
| 소소코드 | | | |
| <첨부>  package com.bit.day16;  import java.io.\*; import java.util.\*; import java.util.function.UnaryOperator;  public class Test extends Thread {  static ArrayList<HashMap<String, String>> data = new ArrayList<>();  static String[] title = { "학번", "이름", "국어", "영어", "수학" };  static File file = new File("data.txt");  static int num = 0;   Scanner sc = new Scanner(System.in);   double avg = 0;   FileWriter fw = null;  FileReader fr = null;   BufferedReader br = null;  BufferedWriter bw = null;   void input() {  num++;   HashMap<String, String> map = new HashMap<>();  map.put(title[0], num + "");  for (int i = 1; i < title.length; i++) {  System.out.print(title[i] + "> ");  map.put(title[i], sc.nextLine());  }  data.add(map);  }   void modifyinput() {  HashMap<String, String> map = new HashMap<>();  map.put(title[0], num + "");  for (int i = 1; i < title.length; i++) {  System.out.print(title[i] + "> ");  map.put(title[i], sc.nextLine());  }  data.set(num - 1, map);  }   void resetsave() {  try {  file.delete();  data.clear();  num = 0;   if (!file.exists()) {  file.createNewFile();  }   fw = new FileWriter(file, false);  bw = new BufferedWriter(fw);   input();   for (int i = 0; i < data.size(); i++) {  HashMap<String, String> map = data.get(i);  avg = (Double.parseDouble(map.get(title[2])) + Double.parseDouble(map.get(title[3]))  + Double.parseDouble(map.get(title[4]))) \* 100 / 3 / 100.0;   bw.write(map.get(title[0]));  bw.newLine();  bw.flush();   bw.write(map.get(title[1]));  bw.newLine();  bw.flush();   bw.write(map.get(title[2]));  bw.newLine();  bw.flush();   bw.write(map.get(title[3]));  bw.newLine();  bw.flush();   bw.write(map.get(title[4]));  bw.newLine();  bw.flush();   bw.write(String.format("%.2f", avg));  bw.newLine();  bw.flush();  }   bw.close();  fw.close();  System.out.println("저장되었습니다!");   } catch (IOException e) {  // TODO Auto-generated catch block  e.printStackTrace();  }  }   void autosave() {  while (true) {  try {  sleep(20000);   if (!file.exists()) {  file.createNewFile();  }   fw = new FileWriter(file, true);  bw = new BufferedWriter(fw);   for (int i = 0; i < data.size(); i++) {  HashMap<String, String> map = data.get(i);  avg = (Double.parseDouble(map.get(title[2])) + Double.parseDouble(map.get(title[3]))  + Double.parseDouble(map.get(title[4]))) \* 100 / 3 / 100.0;   bw.write(map.get(title[0]));  bw.newLine();  bw.flush();   bw.write(map.get(title[1]));  bw.newLine();  bw.flush();   bw.write(map.get(title[2]));  bw.newLine();  bw.flush();   bw.write(map.get(title[3]));  bw.newLine();  bw.flush();   bw.write(map.get(title[4]));  bw.newLine();  bw.flush();   bw.write(String.format("%.2f", avg));  bw.newLine();  bw.flush();  }   bw.close();  fw.close();  System.out.println("저장되었습니다!");   } catch (InterruptedException | IOException e) {  // TODO Auto-generated catch block  e.printStackTrace();  }  }  }   void load() {  try {  fr = new FileReader(file);  br = new BufferedReader(fr);   String line = "";  for (int i = 1; (line = br.readLine()) != null; i++) {  System.out.print(line + "\t");  if (i % 6 == 0) {  System.out.println();  }  }  br.close();  fr.close();  } catch (FileNotFoundException e) {  // TODO Auto-generated catch block  e.printStackTrace();  } catch (IOException e) {  // TODO Auto-generated catch block  e.printStackTrace();  }  }   void modify() {  System.out.print("학번을 입력하세요> ");  String input = sc.nextLine();   try {  fr = new FileReader(file);  fw = new FileWriter(file, true);   br = new BufferedReader(fr);  bw = new BufferedWriter(fw);   String line = "";  for (int i = 1; (line = br.readLine()) != null; i++) {  if (input.equals(line)) {  modifyinput();  }  }   bw.close();  br.close();   fw.close();  fr.close();  } catch (FileNotFoundException e) {  // TODO Auto-generated catch block  e.printStackTrace();  } catch (IOException e) {  // TODO Auto-generated catch block  e.printStackTrace();  }   }   // 전체삭제  void remove() {  file.delete();  data.clear();  int num = 0;   System.out.println("데이터를 전부 삭제했습니다");  }   @Override  public void run() {  autosave();  }   public static void main(String[] args) {  Scanner sc = new Scanner(System.in);  System.out.println("학생성적관리 프로그램(ver 0.6.0)");   Test me = new Test();  me.start();   while (true) {  System.out.print("1. 입력 2. 리셋 후 입력 3. 보기 4. 수정 5. 삭제 0. 종료> ");  int input = Integer.parseInt(sc.nextLine());  if (input == 0) {  me.interrupt();  break;  }  if (input == 1) {  me.input();  } else if (input == 2) {  me.resetsave();  } else if (input == 3) {  if (file.length() != 0) {  System.out.println("-------------------------------------------------");  System.out.println(title[0] + "\t" + title[1] + "\t" + title[2] + "\t" + title[3] + "\t" + title[4]  + "\t" + "평균");  System.out.println("-------------------------------------------------");  me.load();  System.out.println("-------------------------------------------------");  }  } else if (input == 4) { // me.modify();  } else if (input == 5) {  me.remove();  }  }  } } | | | |