Längdskidtävling

- Gruppuppgift




Lärosäte: YrkesAkademin

Program: Systemutvecklare Java

Kurs: Objektorienterad programmering

Grupp: 5A

Medverkande: Edbom Eriksson, A., Lundqvist, A., Ojail, M &, Sjödin, A

Innehåll

Uppgift

Projektet. Längdskidstävlingen

Grundläggande krav

Applikationen ska kunna användas i huvudsak vid längdskidtävlingar för att supporter/ledare ska kunna ta mellantider på åkare längs skidspåret. Längdskidtävlingar tävlas antingen med individuell start där varje åkare startar med 15 sek eller 30 sek mellanrum, genom masstart (dvs. alla åkare samtidigt) eller såsom jaktstart där åkarna startar utifrån resultatet i en föregående tävling.

I dess enklaste form ska applikationen kunna räkna fram mellantiden för ett visst åkarnummer vid den plats på spåret som ledaren står vid. Om till exempel första start är kl. 10:00 så behöver kunna räkna fram hur lång åktidsomstartnummer 1 som startade 10:00:00har, startnummer 2 som startade 10:00:30 och startnummer 3 som startade 10:01:00 etcetera har vid en viss plats på spåret. Om användaren matar in ett startnummer, så ska aktuell åktid samt placeringen i förhållande till de som hittills passerat. Obs: Instruktionen är vag med flit eftersom ni förväntas utöva kravfångst genom att ställa följdfrågor till mig.

Utökad kravspecifikation

Skriftligt ställda frågor och svar

- Ska stafett vara med eller är det endast vanlig längdskidåkning?
    - Svar: Endast vanlig tävling
- Ska det gå att söka efter damer/män/ungdom etcetera
    - Svar: Det behövs inte. Vill man söka på en mellantid, så är det bara intressant mot övriga i klassen.
- Ska man få upp tabeller och kunna spara tidigare starter
    - Svar: Bra är om man kan skriva ut (på skärmen) startlista och resultatlista
- vilka distanser gäller? Både för hela loppet samt mellantider
    - Svar: Distanser behövs inte för en enkel tidtagning.
- är det med km eller miles?
    - Svar: Behövs ej
- Vilka personuppgifter på åkare ska finnas?
    - -fnamn: Ja
    - -enamn: Ja
    - -land: Extra
    - -klubb: Extra
    - -startnummer: Ja
- Har en åkare alltid samma startnummer?
    - Svar: Det är bra om man när man anmält åkare, kan göra en startnummer lottning. Man anger första start, intervall mellan åkare och första startnummer
- hur många lopp räknas in sedan tidigare? Är det endast senaste loppet som avgör eller kan det vara flera?
    - Svar: Bara en tävling
- ska man kunna logga in för att använda tjänsten eller är det öppet för alla?
    - Svar: Alla som kan starta programmet ska kunna köra det. Ingen web ska användas

- ska en tidsdifferens finnas för att jämföra vilken aktuell placering åkaren har och för tidsdifferens mellan åkare?
    - Svar: Om användaren slår in ett startnummer för att kolla mellantid, ska placering av de som hittills passerat och tid efter ledaren visas.
- Hur ska allt sparas? Finns det något krav att det ska vara till en fil tillexempel. ?
    - Svar: Det behöver inte sparas. Datat lagras endast i minnet. Ni kan bygga ut med lagring om ni är klara med övrigt.

Övriga framställda krav

1. Att mata in anmälningar
2. Att lotta startnummer baserat på en första start, intervall och första startnummer per eventuell klass.
3. Skriva ut start och resultatlista på skärmen.
4. Kunna skriva in ett startnummer för att få mellantid (åkt tid) och placering för åkare som passerat.
5. Mål tidtagning.
6. Skriv in startnummer och när användaren trycker ”enter” registreras aktuell tid som målgångstid.
7. Det räcker med individuellstart
8. Programmet behöver bara köras som textbaserat i kommandotolken.
9. Mata in namn o klass Första startnummer, första starttid o intervall den aktuella tiden - starttiden.
10. Ska kunna jämföras med andra åkare
11. Måltidtagning Skriv in hårdkodade åkare (5st tillexempel),
    i. registrera sen några åkare varje gång när programmet startas
    ii. Sen ska det lottas
    iii. När de har gått i mål så skrivs resultatlistan skrivas ut

Hur vi valt att arbeta

Då majoriteten av gruppen är bosatt på olika orter har allt arbete behövt ske över distans. För att detta skulle vara möjligt var det viktigt att se över hur vi skulle lösa det.

Planering

Alla i gruppen samlades i en Discord-grupp och skapade tillsammans ett GitHub-projekt som alla kunde vara delaktiga i. Därefter fick alla möjlighet att tala om när var enskild person kunde sitta och arbeta samt lämna ut sina kontaktuppgifter.

När detta var klart påbörjades diskussionen kring hur projektet skulle se ut, ett flertal varianter av flödesscheman ritades upp och frågeställningar till läraren skickades för att få mer klarhet kring uppgiften.

Efter varje session talade gruppen om en ny tid för att arbeta och valet att skriva själv eller tillsammans.

Kommunikation

Som kommunikationsverktyg har Discord (både i text- samt röstkanal) använts, Zoom och GitHub genom att kommentera varandras kod samt starta olika projekt med ”Att göra listor”.

Grupp/solo

Den största delen av arbetet har skett genom att en person har kodat och de andra har suttit med och diskuterat hur det bör se ut. Utöver det har också det varit tillfällen när en person kommit fram till en möjlig lösning och då startat en egen Branch med kod. Kod har alltid visats för gruppen innan det gått till Main.

Vad har fungerat bra

Alla i gruppen är nöjda över resultatet och stämningen har varit god. Det har varit en bra dynamik där alla har kunnat bidra till något.

Vad har fungerat dåligt

Då gruppen varit ivrig att komma i gång påbörjades arbetet innan svar från kravställaren kommit. Detta gjorde att programmet fick genomgå stora förändringar ett flertal gånger samt att flödesscheman ritades i flera olika upplagor.

Test hantering

Test hanteringen för programmet har skett under hela projektets gång. Både delar av programmet samt programmet i sin helhet testades för att finna eventuella buggar, lösa problem samt få programmet att uppnå en användbarhet.

I slutskedet av programmet testades allting med stor noggrannhet samt med försök att få fram tidigare fel, felhanteringar blev projektets stora pusselbit.

Hur tester genomförts

Hjälpmedel

För att kunna söka igenom fel har projektet körts genom Eclipse Consol samt att kod redovisats i GitHub för en översikt av koden. Word dokument samt GitHubs projekthanteringar har stått för dokumentation av vad som behöver göras och vad som blivit klart.

I Microsoft Word har även färgkodning använts för att få en lättare översikt på hur långt varje felhantering kommit. Grön text visade att det är färdighanterat, gult/orange att det fortfarande behöver mindre ändringar och svart (normal) text att det inte är gjort.

Grupp

De flesta större tester har genomförts genom att hela gruppen har suttit i ett samtal där en person streamat när programmet körs. Därefter har alla tillsammans fått försöka se eventuella fel som uppstått. Tester har även gjorts av alla ansvariga på egen hand genom läsning av kod samt körning av programmet.

Vad som testats

Tidmetoder

Tester som fick mycket fokus var att kontrollera att de tider som gavs ut till åkarna inte krockade för mycket mot varandra. En mellantid får inte tagit längre tid än en sluttid och det fick inte heller vara för stort avstånd mellan alla åkare. Beräkningen för åktiden fick även den stor fokus, det var viktigt att en åkare som startat 30sekunder efter den första åkaren också fick den tiden avdragen när den kommit i mål.

När gruppen insåg att en detalj missats, att det skulle vara möjligt att manuellt lägga in tid för sluttid. Var det tvunget att skriva om koden något.

Genom att köra programmet ett flertal gånger och granskning av kod tillsammans samt uppgiften i sig kom vi till slut fram till en fungerande metod och tankesätt som alltid givit en jämn resultatlista.

Programmet låser sig

En bugg som ibland uppstått är att programmet låser sig, en del av gångerna visade det sig att det saknades en nextLine() och andra gånger visade det sig vara en miss av måsvinge som gjort att fortsättningen av programmet hamnat fel.

För att få en sådan säker felhantering på dessa problem som möjligt testades programmet att köras på flera sätt. Till exempel var en att klicka menyvalen innan de visats, andra att skriva olika outputs i olika ordningar. Ett flertal gånger användes här Zooms funktion att rita för att lättare kunna förklara för varandra vart i koden det skulle kunna vara fel och hur mycket av en funktion som skulle behöva flyttas.

PrintOut

Att rätta stavfel och att bibehålla en grammatik och språk som håller en rödtråd genom hela tävlingen har varit viktigt för att programmet ska upplevas professionellt. Här har alla i gruppen fått gå igenom så det ser bra ut och försökt hjälpa varandra att se eventuella missar.

Utskrift för resultat har kontrollerats så de faller in i rätt kolumn och tester för hur fort all text samt pilar ska visas för att programmet ska flyta på i lagom takt.