# CPROG Rapport för Programmeringsprojektet

Gruppnummer: 17

Gruppmedlemmar:

Robin Justice Akyuz, 200309202356

*Fyll i 'check-listan', så att du visar att du tagit hänsyn till respektive krav, skriv också en kort kommentar om på vilket sätt du/gruppen anser att kravet tillgodosetts, och/eller var i koden kravet uppfylls.*

*Den ifyllda Rapportmallen lämnas in tillsammans med Programmeringsprojektet*. Spara rapporten som en PDF med namnet CPROG\_RAPPORT\_GRUPP\_NR.pdf (där NR är gruppnumret).

## Beskrivning

Jag har gjort en spelmotor med SDL2 och C++, samt implementerat ett basketspel med hjälp av spelmotorn. Under mappen Resources/images finns bilderna till spelaren, motståndaren, basketkorgen och basketbollen som är själva basketplanen. Under Resources / fonts finns arial vilket är fonten som används till score countern i spelet. Det krävs också att du lägger hela resource under build/debug om du är på windows. Samma sätt gäller Include foldern där du behöver lägg till din egna include path.

Spelet handlar om att du styr en basketspelare som är utrustad med en basketboll. Den går att skjuta, där du helst vill träffa någon av basketkorgarna – då får du poäng! Missar man får du inga poäng, och om du träffar motståndaren tar motståndaren 2 av dina poäng. Går du nära motståndaren och nuddar henne så startas spelet om. Dina poäng blir till 0 och du hamnar på startpositionen igen. Spelet har en inbyggd ”recoil” som gör att du som spelare behöver sikta efter varje skott, och den kan inte förutses.

W,A,S,D används för att röra, SPACE för att skjuta

Spelet behövs MAKE:as genom att skriva ”make” i terminalen

Spelas startas när du har skrivit ”./build/debug/play” i terminalen.

## Instruktion för att bygga och testa

Spelet behövs MAKE:as genom att skriva ”make” i terminalen. OBS! här krävs det också att ni har följt beskrivningen om resources.

Spelas startas när du har skrivit ”./build/debug/play” i terminalen.

## Krav på den Generella Delen(Spelmotorn)

* 1. [ Ja] **Programmet kodas i C++ och grafikbiblioteket SDL2 används.**Kommentar: Spelmotorn använder sig av SDL2 för att rita upp och rendera diverse grafik och c++ används för att programmera. Exempelvis datamedlemmar, klasser och funktioner.
  2. [ Ja] **Objektorienterad programmering används, dvs. programmet är uppdelat i klasser och använder av oo-tekniker som inkapsling, arv och polymorfism.**  
     Kommentar: Ja, spelmotorn har diverse klasser som använder sig av arv, i form av basklasser och subklasser. Exempelvis är Component är basklass för alla sprites, där klasserna MovingObj och StillObj ärver från Component klassen. Båda två klasser har också sina egna subklasser i form av ex. Player, Projectile osv. Med det sagt finns det både kort och långa arv för klasser i min spelmotor. De flesta klasser bör ha en protected konstruktor och att man kan skapa en instans av klasser med fabriksfunktioerna ”getInstance”, följt efter objektet (Basketball, Player osv.)
  3. [ Ja ] **Tillämpningsprogrammeraren skyddas mot att använda värdesemantik för objekt av polymorfa klasser.**  
     Kommentar: Ja, det går inte att skapa ett nytt objekt med keywordet ”new” på alla klasser. Det betyder alltså att den som spelare inte kan skapa en ny Component med ”new” utan, som beskrivet ovanför, måste använda sig av klasser med fabriksfunktioner för att skapa instanser av klassen. Subklasser kan med hjälp av den ärvda klassen skapa sig självt.
  4. [ Ja] **finns en gemensam basklass för alla figurer(rörliga objekt), och denna basklass är förberedd för att vara en rotklass i en klasshierarki.**  
     Kommentar: Ja, det finns en klass Component som både rörliga och stilla objekt ärver ifrån.
  5. [ Ja] **Inkapsling: datamedlemmar är privata, om inte ange skäl.**Kommentar: Ja, alla datamedlemmar ligger under den privata sektionen i varje klass, inga finns publika. Ett undantag är den globala medlemmarna för system och session. Det finns även en publik medlemm vid damn: is\_alive som anropas om ett objekt är redo att tas bort.
  6. [ Ja] **Det finns inte något minnesläckage, dvs. jag har testat och sett till att dynamiskt allokerat minne städas bort.**  
     Kommentar: När bollen som skjuts av spelaren hamnar utanför spelplanen, nuddar en motståndaren eller en korg så anropas destruktorn och tar därför bort bollen.
  7. [ Ja ] **Spelmotorn kan ta emot input (tangentbordshändelser, mushändelser) och reagera på dem enligt tillämpningsprogrammets önskemål, eller vidarebefordra dem till tillämpningens objekt.**  
     Kommentar: Spelet tar emot hantering av tangentbordshändelser i Session klassen, under medlemsfunktionen run().
  8. [ Ja] **Spelmotorn har stöd för kollisionsdetektering: dvs. det går att kolla om en Sprite har kolliderat med en annan Sprite.**  
     Kommentar: Ja, Component klassen har två inbyggda funktioner som testar och hanterar kollisionsdetektering.
  9. [ Ja] **Programmet är kompilerbart och körbart på en dator under både Mac, Linux och MS Windows (alltså inga plattformspecifika konstruktioner) med SDL 2 och SDL2\_ttf, SDL2\_image och SDL2\_mixer.**  
     Kommentar: Jag har inte gjort några specifika krav som gäller för någon konsol eller operativsystem, varje sig windows, mac eller linux.

## Krav på den Specifika Delen(Spelet som använder sig av Spelmotorn)

* 1. [ Ja] **Spelet simulerar en värld som innehåller olika typer av visuella objekt. Objekten har olika beteenden och rör sig i världen och agerar på olika sätt när de möter andra objekt.**  
     Kommentar: Spelet simulerar en basketplan med diverse basketkorgar, spelare och hans basketbollar samt motståndare. I bakgrunden finns det åskådare som tittar på (PNG). Basketbollen kan skjutas, spelare och motståndare kan röras och tillsammans simulerar ett spel.
  2. [ Ja] **Det finns minst två olika typer av objekt, och det finns flera instanser av minst ett av dessa objekt.**  
     Kommentar: Det finns två objekt, spelare och motståndarne (player, enemy) som både instanserar sig från Moving Obj. Basketbollen instansieras av objektet varje gång spelaren skjuter.
  3. [ Ja] **Figurerna kan röra sig över skärmen.**   
     Kommentar: Både du och motståndaren rör sig över spelplanen, du i kombination med vilka knappar du trycker samt att spelaren har ett skript.
  4. [ Ja] **Världen (spelplanen) är tillräckligt stor för att den som spelar skall uppleva att figurerna förflyttar sig i världen.**  
     Kommentar: Ja, den som spelar spelet kan uppleva att motståndaren går från höger till vänster enligt ett skript. Samma ska gället för din egna spelare, att man kan se att gubben rör sig efter vilka knappar du tryckt.
  5. [ Ja ] **En spelare kan styra en figur, med tangentbordet eller med musen.**  
     Kommentar: Ja, man kan styra ”player”, alltså basketspelaren med tangentbordet. Med knapparna, w, a, s och d, gör att spelaren respektive går frammåt, backåt, till vänster och till höger. Med mellanslagsknappen (space) kan spelaren också skjut basketbollen.
  6. [Ja] **Det händer olika saker när objekten möter varandra, de påverkar varandra på något sätt.**  
     Kommentar: Om spelaren går mot spelaren, ”reset:as” spelat, alltså att spelaren går tillbaka till sin startposition och poänget ”score” sätts till 0. Om bollen nuddar ett ”hoop” (korg), så får spelaren 2 poäng. Nuddar bollen däremot motståndaren ”Enemy” går poängen ned med -2. I kort kan spelaren både förlora och få poäng genom att skjuta bollen, samt att kan spelet startas om vid kollision med motståndaren.