C++ Tärningspelet

Introduktion

På denna uppgift ska man göra ett tärningsspel. Detta tärningspel går till på att man ska lägga in pengar sedan satsa de. När du har gjort det ska du kasta två tärningar. Efter att du har kastat tärningarna så är det din största tärning mot datorns största tärning. Efter att du har vunnit eller förlorat så får du reda på hur mycket du har på kontot och sedan om du vill spela en gång till eller inte. För att spela så måste du minst ha tillgång till 100 kr på kontot.

Problembeskrivning

Här skriver du om vilka delproblem som du identifierat.
Insättning av pengar
När man sätter in pengar på kontot så ska man kunna sätta in allting mellan 100 och 5000. När man sätter in mindre ska programmet förklara varför du inte kan det och ge dig så många chanser som möjligt tills du sätter in rätt summa.
Satsning
När man har lagt in ett belopp så ska man kunna satsa. Satsningen går ut på att man inte ska kunna satsa mer än vad man har lagt in på kontot. Sen ska man inte kunna satsa något annat än 100,300 eller 500.
Tärningskast
Efter satsningen så ska programmet kasta två tärningar. När man har kastat två tärningar så ska programmet se vilken som är din största tärning och vilken som är datorn största tärningen. Sedan ska båda era största tärningar jämföras och den största tärningen vinner. Det ska spelas bäst av tre och om man vinner två gånger så behövs det ingen tredje omgång. Alla omgångar ska synas tydligt vem som vinner och vem som förlorar.
Resultat
När spelaren har kört klart sin bäst av tre så får han en totalavinstsumma. Denna totalavinstsumman visar om det har gått bra för spelaren eller inte. Så om du kör två gånger och vinner 500 båda gångerna kommer det att stå 1000. Det visas även om du går minus och skulle råka förlora 500 två gånger och i så fall står det -1000.
Avsluta
Spelaren ska få möjligheten till att avsluta efter varje omgång.

Antaganden och krav

Beskriv dina egna antaganden och krav som du identifierat för att kunna lösa uppgiften.
Oavgjort
Vid oavgjort så kommer programmet att ge vinsten till datorn.
Visa pengarna på kontot
På detta program så visas ditt belopp när du har lagt in det och efter att du har spelat färdigt en omgång.
Regler i början
Detta program innehåller även mycket regler innan spelet startar. Man får upp reglerna bara en gång när man startar programmet.
Tröstmedelanden

Om spelaren skulle förlora så får den ett tröstmedeland.
Insättning
Jag valde själv att man inte ska kunna lägga in mindre än 50 då man inte kan satsa 50 och det skulle bara vara slöseri med pengar.

Lösningsdesign

Beskriv hur du tänkt och gått tillväga för att lösa uppgiften. För uppgift 1 anger du även ditt flödesdiagram: en modell som löser uppgiftens problem.
Insättning:
För att lösa problemet måste man först göra en variabel för pengarna. Detta gör man genom att skriva int sedan den variabel namnet och avsluta det med;. När man har gjort en variabel för pengarna så ska man nu förklara för spelaren att sätta in pengar. Detta gör man genom en cout. Cout är en ouput som programmeraren pratar med spelaren. Direkt efteråt så gör man in cin så då är en input. En cin är en input som spelaren kan skriva in ett värde. När man har skrivit in cin så skriver man variabeln för pengarna. Nu bestämmer spelaren hur mycket den vill sätta in på kontot. Nu ska man göra regler för hur mycket man får lägga in på kontot. Detta gör vi med en while sats. En while sats är en slags loop som behöver uppfylla något. Om det som står inte är uppfyllt så kommer det inte att gå ur while loopen. Först måste vi skriva while sedan att om det är mindre än 100 eller om det är mer än 5000 så åker man in i loopen. Ända sättet att gå ur den är att skriva något mellan de talen.
Satsning:
Nu när går det att lägga in pengar på programmet. Nu ska det kunna gå att satsa. Det gör vi på samma sätt som att lägga in pengar. Vi gör en cout på att nu kan du satsa mellan 100, 300 eller 500. Sedan gör vi en cin där använder får skriva in hur mycket den vill satsa. Nu måste vi göra regler om använder skulle skriva in fel värde. Första kravet är om använder satsar mer än vad som ligger på kontot. Då behöver vi göra en while loop. Man kommer in på denna while loop om man skriver in en satsning mer vad som är på kontot. Då kommer du få ett meddelande om att du har satsat för mycket för det som ligger på kontot. Sedan får du en chans att skriva in rätt med en cin. Du kommer återigen inte ur while loopen om du inte skriver in rätt summa som då är lika mycket eller mindre än det du har på kontot. Sedan har vi ett till krav på att du inte får betta något annat än 100, 300 eller 500. Detta gör du med hjälp av att göra en till while loop. Du kommer in på denna while loop om du satsar något annat än 100, 300 eller 500. Detta gör du med att skriva bet=!100 bet=!300 bet=!500. Detta betyder om satsningen inte är lika mycket med 100,300 eller 500 så kommer den att gå in i while loopen tills du skriver in rätt. Där har jag återigen skrivit cout på vad du har gjort för fel och sedan en cin som du kan rätta den på.
Tärningskast:
Nu är man på den delen som tärningen ska bli kastad på. Först gjorde jag variabler för mina två tärningar och för de andra två. Efter det så du även skapa två till variabla för högstaspelartärningen och högstadatortärningen. Sedan skrev jag rand() % 6+1. Detta gör så att programmet väljer ett random tal mellan 6 och 1. Detta gjord jag på alla fyra tärningar. Längre uppe på programmet måste man srand(time(0)) för att datorn ska ha något värde att utgå ifrån när den skriver ut slumpmässiga tal. Nu ska vi få reda på vilken av spelaren tärningar är störst och vilken av datorn tärningar är störst. Det gör man med hjälp av en if else sats. If satsen kommer att behålla ett bestående. Om det bestående är sant så

kommer programmet att välja den if satsen och allt som står innanför den satsen kommer att göras. Om det inte stämmer kommer den att hoppa till else satsen som kommer att vara allt annat. Efter det ska man skriva högstaspelartärningen=tärning1 på if satsen och högstaspelartärningen=tärningen2 på else satsen för att man ska kunna jämföra datorn och spelarens poäng i slutet. Nu ska man göra exakt likadant på datorns tärningar där man får ut den största tärningen där. Efteråt så jämnför man de två tärningarna på en if sats. Nu använder vi if och två else if satser. Den första if är om spelarens träning är större än datorns tärning. Om det inte stämmer så kommer det ner till else if där om datorns tärning är större än spelarens tärning. Tillslut har vi den sista else if där påståendet är om datorntärning och spelarens tärning är lika mycket. Nu skriver man vad som händer om de if satserna blir sanna. Innan de så måste man göra spelarepoäng och datorpoäng variabler. Efter det så skriver du if satserna om spelarens tärning är större än datorn så får spelaren en poäng men om datorns tärning är större så får datorn en poäng och om det blir lika så får datorn en poäng. Det vi har gjort nu är att vi bestämmer vilken som är den största spelartärningen och datortärningen sedan tar vi deras två största och jämför de. När vi har gjort det så ger vi poäng till spelaren vid vinst och en poäng till datorn vid vins eller lika. Nu är det som saknas att göra en while loop runt om spelet så att det hinner bli en bäst av tre. Detta gör man med att skriva en while loop längst upp där påståendet är om scorepoängen och datorpoängen inte är lika med 2. Då kommer denna loop hålla på tills någon har 2 poäng och där med vunnit bäst av tre. Efter att någon har vunnit bäst av tre så skriver man en if sats efteråt. OM spelaren har vunnit och en else där datorn har vunnit. Om spelaren har vunnit så får spelaren pengarna på kontot och bettet. Det skriver man så här: konto=konto+bet och om man förlorar så står det så här på if satsen: konto=konto-bet.
Resultat:
Nu ska man försöka få fram en vinstsumma som alltid stämmer oavsett hur många omgångar spelaren bestämmer sig att spela. Detta gör man med att skapa en ny variabel. Sedan skrivs det på if satsen som under tärningskastt while loopen skrivs totalvinst= totalvinst+bet och på else skrivs totalvinst=totalvinst-bet. Detta kommer att räkna ut totalvinstsumman oavsett om man förlorar eller vinner. Något att notera är att det kan gå minus om spelaren skulle förrlora mer än vad den vinner. Sedan ska det skrivas ut till spelarne att se. Det gjorde jag med en cout efter räkningen. Där skriver jag. Din totalvinstsumma <<totalvinst. Detta kommer att visa totalvinstsumman direkt efter spelomgången är slut.
Avsluta:
Nu när spelet fungerar och man får fram en vinnare så återstår det att spelet loopar och att det går att avsluta det efter varje omgång. Jag gjorde det med hjälp av en bool. En bool kan bara ha två värden, true eller false. Först skapades det en bool utanför while loopen som har värdet true. Sedan ska man göra en while loop om boolen har en true värde. Sedan skapades det en if sats efter att spelaren har lagt till mer pengar efter omgången är slut. Där finns det en ny variabel som man ska ge ett värde genom cin >>. Om spelaren skriver 1 så kommer man in på en if sats. Om det stämmer och den tar första if satsen så står det att boolen har ett true värde. Eftersom att man endast kommer in på while loopen om värdet på boolen är true så kommer spelet att startas om, men eftersom att spelaren redan har lagt in pengar så har kommer att else satsen också ha bool satsen true. Detta gör möjligheten för spelaren att fortsätta även om den skulle skriva fel nummer.

Diskussion

Diskutera din lösning genom att identifiera dess styrkor och brister.
Jag tycker att jag har skapat ett fungerande spel men med tyvärr flera brister. Den första bristen på programmet är att man kan enkelt komma förbi några while loopar när man ska satsa pengar. Jag har en som försäkrar att man har mer än vad som finns på kontot och en annan som säger åt spelaren att bara kunna satsa 100,300 eller 500. Tyvärr så går det att lägga in exempelvis 300 kr och sedan testa satsa 150 för att komma förbi första while satsen för att sedan kunna satsa 500 när man inte har det på kontot.

Källkod

Bifoga din källkod i rapporten. Formatera den så att det går att följa koden. Ange kommentarer i koden som tydligt beskriver dess avsikt. Observera att koden måste överensstämma med flödesdiagrammet.

```
using namespace std;

int ScorePlayer;

int ScoreComputer;

int main() {
setlocale(LC_ALL, "swedish");

int totalvinst = 0;

int inpengar;

cout << "Hej och välkommen till träningsspelet, här kommer du kunna lägga in pengar mellan
100 till 5000 och satsa 100,300 eller 500. När du har satsat kommer du och datorn kasta
tärninger, när tärningarna är kastade så kommer din högsta gå emot datorns högsta för att
utse en vinnare. " << endl;

cout << "lägg in pengar: ";

cin >> inpengar;                                    // lägger in pengar

while (inpengar < 100 || inpengar > 5000) {       // Regel för att man inte får lägga in
mer än 5000 och inte mindre än 100, annars skriver du in tills det är rätt.

cout << "Du har lagt in fel summa pengar, var vänlig och förösk igen." << endl;

cin >> inpengar;

}



cout << "Du har: " << inpengar << endl;

int back1;
back1 = 0;
bool back = true;
```

```
while (back == true)                        // gör en loop runt hela spelet om back är true
och det kommer den att vara så länge man vill spela vidare
            {
int bet;

cout << "Betta mellan 100, 300 eller 500: ";

cin >> bet;                                          // Lägg in bet



while (inpengar < bet) { // gör en regel för att inte kunna betta mer än vad som finns på
kontot, annars skriver du in tills det är rätt.

cout << "Du har inte tillräckligt med pengar för att betta den summan." << endl;

cout << "betta rätt summa";

cin >> bet;
}

while ((bet != 100) && (bet != 300) && (bet != 500)) // gör en regel för att inte kunna
betta något annat än 100,300 eller 500 , annars skriver du in tills det är rätt.


                          {
cout <<" fel bet" << endl;

cout << " Betta igen: ";

cin >> bet;
}


cout << "Du har bettat: " << bet << endl; // visar hur mycket du har bettat

srand(time(0));

int highestdiceplayer;
int highestdicecomputer;

ScoreComputer = 0;
ScorePlayer = 0;
while (ScoreComputer != 2 && ScorePlayer != 2) {           // gör en while loop runt
själva tärningskastet tills spelaren eller datorn får två poäng

int slumptal = rand() % 6 + 1;

int slumptal2 = rand() % 6 + 1;

int slumptal3 = rand() % 6 + 1;

int slumptal4 = rand() % 6 + 1;


cout << "Nu kastar spelaren tärnigen" << "\n" << endl;// här får vi reda på hur mycket
spelarens två tärningar blev
```

```
cout << "Tärning1: " << slumptal << endl;

cout << "Tärning2: " << slumptal2 << endl;


if (slumptal >= slumptal2) { // På denna IF sats får vi reda på vilken tärnigns som är
störst

// sedan ger vi den största tärningen en ny variabel

cout << "Din högsta tärning: " << slumptal << endl;
highestdiceplayer = slumptal;
}
else {
cout << "Din högsta tärning: " << slumptal2 << endl;
highestdiceplayer = slumptal2;

}

cout << "Nu kastar datorn tärningen" << "\n" << endl;

cout << "Tärning1: " << slumptal3 << endl;
cout << "Tärning2: " << slumptal4 << endl;

if (slumptal3 >= slumptal4) { // På denna IF sats får vi reda på vilken tärnigns som är
störst
// sedan ger vi den största tärningen en ny variabel

cout << "Datorns högsta tärning: " << slumptal3 << endl;

highestdicecomputer = slumptal3;
}

else {

cout << "Datorns högsta tärning: " << slumptal4 << endl;

highestdicecomputer = slumptal4;


}


if (highestdiceplayer > highestdicecomputer) {  // PÅ denna IF sats ser vi vem som vann
rundan av de högsta tärningarna.

//Sedan ger vi de en poäng beronde på vem som vann rundan
cout << "grattis du vann denna runda" << endl;
ScorePlayer++;
}

else if (highestdicecomputer > highestdiceplayer) {
                                                    cout << "tyvärr förlorade du denna
runda" << endl;
                                                    ScoreComputer++;
                                        }

                                        else if (highestdicecomputer == highestdiceplayer) {
```

```
cout << "tyvärr förlorade du denna runda" << endl;
ScoreComputer++;
}

}

if (ScorePlayer == 2) { // På den här if satsen så ger vi vinsten till spelaren ifall den
vann, om inte så tar vi ut den summan den förlorade

// Sedan så räknar vi ut totalvinsten för spelets omgång.
cout << "grattis du har vunnit: " << bet << endl;
inpengar = inpengar + bet;
totalvinst = totalvinst + bet;
}
else {

inpengar = inpengar - bet;
cout << "tyvärr du förlorade: " << bet << endl;
totalvinst = totalvinst - bet;
}

cout << "Din totalvinstsummar är" << totalvinst << endl;

int spelaigen;
int merpengar;
merpengar = 0;

cout << "du har: " << inpengar << " på kontot" << endl;

cout << "vill du spela igen? (1 för ja eller 2 för nej)"; // här får du ett förfrågan om
du vill spela igen, om man trycker på ja kommer man in på If satsen

cin >> spelaigen;
if (spelaigen == 1) {

if (inpengar >= 100)      // På denna if sats så frågar programmet den som har 100 eller
på på kontot om den vill lägga till pengar, om den har mindre så är det ett måste
                                        {
cout << "vill du lägga till mer pengar";
cin >> merpengar;
while (merpengar < 0 || merpengar > 5000) {   // om spelaren skulle redan ha mer än vad
som behövs så kan den skriva 0 så kommer den vidare

cout << "Du har gjort fel" << endl;

cin >> merpengar;
}

}

else if (inpengar < 100)
{
cout << "du måste lägga till mer pengar ifall du vill betta igen" << endl;

while (merpengar < 100 || merpengar > 5000) {

cin >> merpengar;
                                                    }
```

```
}

inpengar += merpengar;
cout << "tryck på någon siffra för att fortsätta:  ";

cin >> back1;

if (back1 == 1)  // OM du skriver 1 eller någon  annan siffra så kommer du att skickas upp
till while loooopen där du får börja om igen om du vill spela vidare
{
back = true;

}
else
{
back = true;
}
}


else {
cout << "tack för att du spelade" << endl; // Den är är else om du inte vill spela vidare.
OM du inte tryckte på 1 längre upp på if satsen så avslutas spelet.
return 0;
}
}
}
```

Eventuellt referenser

Föreläsning 3 och 4 med Christian Lennerholt.