Bilaga A

Programkod

head.h

```
#ifndef _MASTERM_
#define _MASTERM_

#include <iostream>
#include <cstdlib>
#include <iomanip>
#include <cctype>
#include <fstream>
#include <string>
#include <ctime>
using namespace std;

//TAL_STORLEK får maximat vara 10 minst 0
const int TAL_STORLEK = 4;

//Filnamn för att spara HighScore-data
const char HIGHSCORE_DATA[] = "highscore.dat";

//Max platser i highscorelistan
const int MAX_HIGHSCORE = 20;

//Max antal som visas i highscore efter spelomgång
const int MAX_HIGHSPEL = 5;

//Namnstorlek
const int NAMN_STORLEK = 20;

struct score
{
  char namn[NAMN_STORLEK];
  int poang;
  time_t date;
};

int mataIn();
int mnuGetPrint();
void spelaMM(int &totplats, score scores[]);
void printHelp();
void avsluta();
void slumpaTal(int a[]);
bool isOnlyDigits(const string str);
int convertToInt(char charToConvert);
int taEmotGissn(int a[]);
int addScore(score scores[], const int points, int totplats);
void showScore(const score scores[], const int totplats);
void skrivTillFil(const score High[], int ant);
void lasFranFil(score High[], int &ant);
bool kontrollera(int giss[], int hemligt[]);
void highRens(int &ant);

typedef int tal[TAL_STORLEK];

#endif
```

main.cpp

```
#include "head.h"


//Ser till att inmatat värde endast kan anta siffra
//Pre: True
//Post: Returnerat inmatad siffra

int mataIn()
{
  char buffer[100];
  int varde;
  bool felKoll = false;
  do
    {
      if(felKoll == true)
        cout << "Felaktikt värde! Försök igen: ";
      cin.getline(buffer, 100);
      if(buffer[0]=='0')
        return 0;
          felKoll = true;
    }while(!(varde=atoi(buffer)));
  return varde;
}

//Skriver ut meny till programmet
//Pre: True
//Post: Returnerat användarens val
int  menuGetPrint()
{

  cout << " \n\n\n      MENY \n==================="
       << "\n1. Spela \n2. Visa HighScore \n3. Hjälp \n4. Rensa HighScore-listan"
       << "\n0. Avsluta\n\n\n"
       << "Ange ditt val genom att trycka en siffra: ";

  return mataIn();

}

//Startar en omgång av MasterMind
//Pre: totplats >= 0, scores[] måste innehålla totplats antal platser
//Post: Kört en omgång av MasterMind
void spelaMM(int &totplats, score scores[])
{

  int gissning[TAL_STORLEK];
  int hemligt[TAL_STORLEK];
  int antalForsok = 0;
  bool klar = false;
  int avbryt;

  slumpaTal(hemligt);
```

```
  cout << endl
       << "SPELA MASTERMIND!" << endl
       << "=================" << endl << endl
       << "Ett hemligt tal har skapats, gissa det!" << endl
       << "Skriv när som helst EXIT för att avsluta denna spelomgång!" << endl
       << endl;

do
  {
    cout << "Gissa (" << TAL_STORLEK << " siffror): ";

    avbryt = taEmotGissn(gissning);

    if(avbryt != 0)
      klar = kontrollera(gissning, hemligt);

    antalForsok++;

  }while((klar == false) && (avbryt != 0));

 if(klar == false)
   {
     cout << "Det hemliga talet var: ";
     for(int i = 0; i < TAL_STORLEK; i++)
        cout << hemligt[i];
     cout << endl;
   }

 if(avbryt == 0)
   antalForsok = -1;

 if(antalForsok != -1)
   totplats = addScore(scores, antalForsok, totplats);

 if((MAX_HIGHSPEL + 1) > totplats)
   showScore(scores, totplats + 1);
 else
   showScore(scores, MAX_HIGHSPEL + 1);

}

//Visar spelregler för MasterMind
//Pre: True
//Post: Skrivit ut spelregler på skärmen
void printHelp()
{

  cout << endl << "SPELREGLER" << endl
       << "==========" << endl << endl
       << "Spelet slumpar fram ett hemligt tal." << endl
       << "Sedan har Du ett antal försök på Dig att gissa rätt tal genom att\n efter
varje gissning"
       << " få veta vilka siffror som är rätt och rätt" << endl
       << " placerade och vilka siffror som är rätt men fel placerade." << endl
       << "  * Rätt placerade siffor kommer att markeras med ett 'R'. " << endl
       << "  * Om siffran finns med, fast på fel plats, markeras det med ett 'S'. "
<< endl
```

```
      << "  * Siffror som inte finns med i det hemliga talet kommer att markeras
med '_'. " << endl  << "LYCKA TILL!" ;

}

//Skriver tack-text.
//Pre: True
//Post: Skrivit ut tack-text på skrämen
void avsluta()
{

  cout << "Tack för att du ville spela MasterMind!" << endl
       << "Hoppas du vill spela igen snart." << endl;

}

//Rensar HighScore-listan från samtliga inlägg
//Pre: True
//Post: ant är satt till 0, HighScore-listan "tömd"
void highRens(int &ant)
{
  ant = 0;


  cout << endl << "RENSA HIGHSCORE" << endl
       << "===============" << endl << endl
       << "HighScore-listan har rensats!" << endl;
}
```

```
int main()
{
  int Val_men;
  int highscoreplatser;
  score highscore[MAX_HIGHSCORE + 2];

  cout << "Välkommen till M a s t e r M i n d!! " << endl
       << "               ==================== " << endl << endl
       << "Skrivet av: Henrik Bäck               |  Anders Ellvin" << endl
       << "            henrback01@student.kau.se |  andeellv01@student.kau.se";

  srand(time(0));

  lasFranFil(highscore, highscoreplatser);

  do
    {

      Val_men = menuGetPrint();

      switch(Val_men)
        {
        case 1: spelaMM(highscoreplatser, highscore);
          break;
        case 2: showScore(highscore, highscoreplatser + 1);
          break;
        case 3: printHelp();
          break;
        case 4: highRens(highscoreplatser);
          break;
        case 0: avsluta();
          break;
        default: cout << "Fel val, detta val är ej tillgängligt.\n";
          break;
        }

    }while (Val_men != 0);

  skrivTillFil(highscore, highscoreplatser);

  return 0;


}
```

gissa.cpp

```
#include "head.h"


//isOnlyDigits: Funktionen kontrollerar om inskickad sträng består av
//siffror. Negativa tal godkänns.
//pre: true.
//post: returnerar true om strängen bara består av siffror, annars false.
bool isOnlyDigits(const string str)
{
```

```
   for(int i = 0; i < str.length(); i++){
        if(!isdigit(str[i]))
                return false;
   }
   return true;
}

//convertToInt: Funktionen konverterar en sträng bestående av siffor till
//ett heltal.
//pre: isOnlyDigits(stringToConvert) == true.
//post: returnerar heltalsvärdet av strängens siffror.
int convertToInt(char charToConvert)
{
  return charToConvert - '0';
}


//Tar emot gissat tal från användare
// Ser till inmatningen endast kan anta siffror samt är av rätt antal.
//Pre: TAL_STORLEK > 0, a[] har TAL_STORLEK platser
//Post: a[] har blivit tilldelat TAL_STORLEK antal siffror från användare
//      samt returnerat skillnaden mellan imatat värde och sträng "EXIT".
int taEmotGissn(int a[])
{
  char gissn[100];
  bool alltOK = true;
  int avsluta;

  gissn[TAL_STORLEK] = '\0';
  cin.getline(gissn, 100);
  avsluta = strcmp(gissn, "EXIT");

  if(avsluta != 0)
    {
      do
        {
          //Kontrollerar så att det bara är tal inmatade
          while((!(isOnlyDigits(gissn)) || (alltOK == false)) && (avsluta != 0))
            {
              cout << "Inte ett gilltig gissning, endast " << TAL_STORLEK
                   << " siffror, försök igen: ";

              gissn[TAL_STORLEK] = '\0';
              cin.getline(gissn, 100);
              alltOK = true;
              avsluta = strcmp(gissn, "EXIT");
            }

          if(avsluta != 0)
            {
              alltOK = true;

              //Sätter till ogitigt värde
              for(int k = 0; k < TAL_STORLEK; k++)
                a[k] = -1;

              //För över siffror från inmatade värden
```

```
            for(int i = 0; i < TAL_STORLEK; i++)
              {
                a[i] = convertToInt(gissn[i]);
                if(a[i] < 0)
                  alltOK = false;
              }
            if(gissn[TAL_STORLEK] != '\0')
              alltOK = false;
          }
        }while(((alltOK == false) || (gissn[TAL_STORLEK] != '\0'))
                      && (avsluta != 0));
    }
  return avsluta;
}
```

highscore.cpp

```
#include "head.h"

//Lägger till person till highscorlistan.
//Lägger personen på rätt plats (stigande ordnat på poäng)
//Pre:  points >= 0, totplats >= 0.
//Post: Namn, Poäng och Tid har laggts till i score på rätt plats.
//       antal platser i highscoren returnerat (max MAX_HIGHSCORE)
int addScore(score scores[],const int points, int totplats)
{
  int plats = totplats;
  char* tid;
  char buffert[100];

  for(int i = 0; i<totplats; i++)
    if((scores[i].poang > points) && (plats==totplats))
      plats = i;

  if(plats < MAX_HIGHSCORE)
    {

      for(int i = totplats; i>=plats; i--)
          scores[i+1]=scores[i];

      cout << "GRATTIS! Du kom in på HighScore!!!"  <<  endl
           << "Skriv in ditt namn (max " << NAMN_STORLEK  <<  " tecken ): ";
      cin.getline(buffert,100);

      for(int i = 0; i < NAMN_STORLEK - 1 ;i++)
        scores[plats].namn[i] = buffert[i];

      scores[plats].poang = points;
      scores[plats].date = time(NULL);

      if(totplats < MAX_HIGHSCORE)
        totplats++;

    }
  else
```

```
    cout << "Du kom inte in på HighScore-listan, gör gärna ett nytt försök!" <<
endl;

  return totplats;

}

//Visar de bästa placeringarna i highscore (från 1 till MAX_HIGHSCORE)!
//Pre: scores[] har totplats antal poster.
//Post: Highscorelistan har skrivits ut.
void showScore(const score scores[], const int totplats)
{

  int i = 0;

  cout << endl
       << " ** HIGHSCORE ** " << endl
       << "=================" << endl << endl;

  cout << setiosflags(ios::left) << setw(8) << "Plats" << setw(NAMN_STORLEK)
       << "Namn" << setw(10) << "Poäng" << "Tid" << endl << endl;;

  for(i = 0; (i < totplats - 1) && (i < MAX_HIGHSCORE); i++)
    cout << setw(8) << i + 1 << setw(NAMN_STORLEK) << scores[i].namn << setw(10)
         << scores[i].poang << ctime(&scores[i].date) << endl;

  if(i == 0)
    cout << "Ingen har ännu spelat MasterMind, gör ett försök!";

}
```

kontroll.cpp

```
#include "head.h"

//Kontrollerar gissningen och skriver ut vägledning
//Pre:  giss[] och hemligt[] har TAL_STORLEK antal värden
//Post: Vägledning för gissning utskriven samt returnerat
//      true om användaren gissat rätt, annars false
bool kontrollera(int giss[], int hemligt[])
{

  bool retur = true;
  char vagledning[TAL_STORLEK + 1];

  vagledning[TAL_STORLEK] = '\0';

  for(int i = 0; i < TAL_STORLEK; i++)
      vagledning[i] = '_';

  for(int i = 0; i < TAL_STORLEK; i++)
    {
      for(int k = 0; k < TAL_STORLEK; k++)
        {
          if(giss[i] == hemligt[k])
            vagledning[i] = 'S';
        }
      if(giss[i] == hemligt[i])
        vagledning[i] = 'R';
      else
        retur = false;
    }

  cout << "Vägledning:        "  << vagledning << endl << endl;

  return retur;

}
```

save_open.cpp

```
#include "head.h"

//Sparar highscore-listan till fil.
//Pre: ant >= 0, High[] har ant antal poster.
//Post: Skrivit ant HighScore poster till filen HIGHSCORE_DATA.
void skrivTillFil(const score High[], int ant)
{
  ofstream skrivFil(HIGHSCORE_DATA, ios::out);
    if(!skrivFil)
      cout << endl << "Det gick inte öppna filen för att spara data";
    else
      {

        for(int i = 0; i<ant; i++)
          {
```

```
            skrivFil.write(reinterpret_cast<const char * > (&High[i]),
                           sizeof(score));

          }

        skrivFil.close();

      }
}

//Läser in hela highscorelistan från fil
//Pre: ant >= 0, High har lediga platser
//Post: HighScorelistan har lästs in till High och ant har blivit tilldelat antal
lästa poster.
void lasFranFil(score High[], int &ant)
{

  int i = 0;
  ant = 0;

  ifstream lasFil(HIGHSCORE_DATA, ios::in);
    if(!lasFil)
      cout << endl  <<  "Det gick inte öppna filen för att öppna data";
    else
      {
        while(lasFil.peek()!= EOF)
          {
            lasFil.read(reinterpret_cast<char * > (&High[i]), sizeof(score));
            i++;
            lasFil.seekg((i)*sizeof(score));

          }
        lasFil.close();
        ant = i;
      }
 }
```

slumpa.cpp

```
#include "head.h"

//Slumpar ett tal med lika många siffror som TAL_STORLEK
//Pre: TAL_STORLEK <= 10, vektor har TAL_STORLEK platser
//Post: Genererat ett slumptal med TAL_STORLEK antal siffror
void slumpaTal(int a[])
{

  int temp[10] = {0,1,2,3,4,5,6,7,8,9};
  int i = 0, ant = 10, slump;
  while(TAL_STORLEK > i)
    {

      slump = rand() % ant;
      a[i] = temp[slump];
```

```
      temp[slump] = temp[ant - 1];

      ant--;
      i++;

    }

}
```

makefile

```
#Makefil till MasterMind
prog: gissa.o highscore.o main.o save_open.o slumpa.o kontroll.o
       g++ -o MasterMind gissa.o highscore.o main.o save_open.o slumpa.o kontroll.o

gissa.o: gissa.cpp head.h
       g++ -c gissa.cpp

highscore.o: highscore.cpp head.h
       g++ -c highscore.cpp

main.o: main.cpp head.h
       g++ -c main.cpp

save_open.o: save_open.cpp head.h
       g++ -c save_open.cpp

slumpa.o: slumpa.cpp head.h
       g++ -c slumpa.cpp

kontroll.o: kontroll.cpp head.h
       g++ -c kontroll.cpp
```