Bilaga A
Paging

Henrik Bäck
850611-6253
Mathias Andersson
850424-6292

Operating Systems DAVB01
VT 2005

Handledare: **Nils Dåverhög**
Hans Hedbom
Thijs Jan Holleboom

Inlämnad **2007-02-12**

```
#include <iostream>
#include <cstring>
#include <cstdlib>

using namespace std;

#define ord char
#define page 4
#define memSize 8
#define fileSize 400

const int logicMem = fileSize/page;

ord physicMem[memSize][page];
bool physicMemO[memSize] = {0};

struct tabelI{

  int frame;
  bool valid;

};

tabelI pageTabel[logicMem];

int numPage = 0;
char fileName[100];

//Ger bästa plats i minnet
//Pre:  True
//Post: Returnerat bästa plats
int gePlats()
{

  int returen;

  returen = numPage % memSize;

  for(int i = 0;i<logicMem;i++)
    {

      if(pageTabel[i].frame == returen && pageTabel[i].valid == true)
        pageTabel[i].valid = false;

    }

  numPage++;

  return returen;

}

//Returnerar ett ord ur minnet
//Pre:  sida är inom logisk minnesintervall, 0 < offset < 3
//Post: ordet på sidan sida och plats offset har returnerats
ord hamtaIn(int sida, int offset)
{
```

```
  ord returData;
  FILE* inData;
  int adress;

  if(pageTabel[sida].valid == true)
    {

      returData = physicMem[pageTabel[sida].frame][offset];

    }
  else
    {

      inData = fopen(fileName,"r");

      if(inData != NULL)
        {

          if(fseek(inData,sida*page,SEEK_SET) == -1)
            {
              cout << "Sökfel, programmet termineras.\n\n";
              exit(-1);

            }

          adress = gePlats();

          if(fread(&physicMem[adress],page,1,inData) < 1)
           {
             cout << "Läsfel, programmet termineras.\n\n";
             exit(-1);
           }

          physicMemO[adress] = 1;

         pageTabel[sida].valid = true;
         pageTabel[sida].frame = adress;


         returData = physicMem[pageTabel[sida].frame][offset];


        }else{

          cout << "Fel vid filöppning, programmet termineras.\n\n";
          exit(-1);

        }

    }

  return returData;

}
```

```
//Returnerar sida och offset för ord
//Pre:  0 <= ordnr < fileSize
//Post: fyll[0] innehåller sida, fylld[1] innehåller offset.
void getPageOffset(int ordnr, int fyll[])
{

  int sida = 0;
  int offset = 0;

  offset = ordnr % page;

  sida = ordnr / page;

  fyll[0] = sida;
  fyll[1] = offset;


}
//Skriver ut sidtabellen
//Pre: True
//Post: Skrivit ut sidtabellen till skräm
void printPageTable()
{

  cout << "Index \t Frame \t Valid\n\n";

  for(int i=0;i<logicMem;i++)
    {

      cout << i << " \t " << pageTabel[i].frame << "\t" << pageTabel[i].valid
           << "\n";

    }

  cout << "\n";

}

int main()
{

  int val;
  int ordet;
  int values[1];

  cout << "Minnesprogrammet\n================ \n\n";

  cout << "Ange fil att exekvera: ";
  cin >> fileName;

  do
    {
       cout << "Vad vill du göra?\n  1. Läsa hela filen\n  2. Läsa vald
               instruktion\n  3. Visa PageTable \n  0.  Avsluta\nAnge val: ";
      cin >>  val;
```

```
      if(val == 2)
        {
          do{
                  cout << "Ange position[1 - " << fileSize << "]: ";
                  cin >> ordet;
          }while(ordet > fileSize || ordet <= 0);

          ordet = ordet - 1;

          getPageOffset(ordet,values);
          cout << "Ord: " << hamtaIn(values[0],values[1]) << "\n\n";
        }
      else if(val == 1)
        {
          cout << "\n";
          for(int i=0;i<fileSize;i++)
            {
              getPageOffset(i,values);
              cout << hamtaIn(values[0],values[1]);
            }
        }
      else if(val == 3)
        {
          printPageTable();
        }
      else if(val == 0){}
      else
        {
          cout << "Felaktigt val \n\n";
        }
    }while(val != 0);

    return 0;

}
```