```
/*
* DBMS Implementation
*/

#ifndef RECORDPTR_H
#define RECORDPTR_H

#include <sys/types.h>
#include "dbmsproj.h"

// location of a specific record in the buffer
// block is the index number of the block the record is in
// record is the index number of the record in the entries table

typedef struct {
    unsigned int block;
    unsigned int record;
} recordPtr;

// struct for making linked lists of recordPtrs

struct linkedRecordPtr {
    recordPtr ptr;
    linkedRecordPtr *next;
};

//overloading operators for use with struct recordPtr
inline bool operator==(const recordPtr &ptr1, const recordPtr &ptr2) {
    if (ptr1.block==ptr2.block && ptr1.record==ptr2.record){
        return true;
    }
    else return false;
};
inline bool operator<(const recordPtr &ptr1, const recordPtr &ptr2) {
    if (ptr1.block == ptr2.block) {
        if (ptr1.record < ptr2.record)
        {
            return true;
        }
        else
        {
            return false;
        }
    }
    else
    {
        if (ptr1.block < ptr2.block)
        {
            return true;
        }
        else
        {
            return false;
        }
    }

}
inline bool operator<=(const recordPtr &ptr1, const recordPtr &ptr2)
{
    if (ptr1 < ptr2 || ptr1 == ptr2) { return true; }
    else return false;
}

inline recordPtr newPtr(recordPtr ptr,unsigned int offset,unsigned int size) {
    recordPtr result;
    result.block = ptr.block + offset / size;
    int rest = offset % size;

    if (ptr.record + rest >= size) {
        result.block += 1;
        result.record = ptr.record + rest - size;
    }
    else {
        result.record = ptr.record + rest;
    }
    return result;
}

inline recordPtr newPtr(unsigned int offset,unsigned int size) {
    recordPtr zero;
    zero.block = 0;
    zero.record = 0;
    return newPtr(zero,offset,size);
}

// increases the ptr so that it points to the next record
// if pointing at the end of a block, moves to the start of the next

inline void incr(recordPtr &ptr,unsigned int size) {
    if (ptr.record < size - 1) {
        ptr.record += 1;
    } else {
        ptr.record = 0;
        ptr.block += 1;
    }
}

// decreases the ptr so that it points to the next record
// if pointing at the start of a block, moves to the end of the previous

inline void decr(recordPtr &ptr,unsigned int size) {
    if (ptr.record > 0) {
        ptr.record -= 1;
    } else if (ptr.block > 0) {
        ptr.record = size - 1;
        ptr.block -= 1;
    }
}

#endif
```