**Midterm C Coding Example**

Show **C** code for the following function:

int **lastSynonym** (FILE \*pFile, int iPrimaryRBN, int iRecSize)

returns the RBN for the last synonym that begins with the specified primary RBN. It returns 0 if there isn't a primary row. If there is only a primary, it returns its RBN which will be the same as the parameter; otherwise, it returns the RBN of the last synonym on the overflow chain for the specified primary RBN. See the EXTRAS for sample data.

Assumptions:

* See the **EXTRAS** for the typedef of the records in the Hashed File.
* Parameter iPrimaryRBN is the hash relative block number in the primary area..
* Parameter iRecSize is the size of each record in the file
* Parameter pFile is the Hashed File which is already open. You must NOT close it.
* You do **not** have to show any include files.

You must use **fseek(**file, lRBA, iSeekMode**)** and **fread(**psbBuffer, lSizeOneRec, lNumberOfRec, file**),**  in your solution. You may not simply use readRec from program #1.

In the EXTRAS

typedef struct

{

int iNextChain; // RBN of the next synonym in the chain. 0 is the end value

char szKey[20]; // key value

Data data; // the rest of the data

} Record;

Sample Data:

|  |  |  |  |
| --- | --- | --- | --- |
| RBN | **Next Chain** | **Key** | **Data** |
| 0 | Header Record | | |
| 1 | 0 | 000000000 | 00000000000000000000000000000000 |
| 2 | 50 | FUN222 | data … |
| 3 | 0 | ONE333 | data … |
| 4 | 0 | 000000000 | 00000000000000000000000000000000 |
| 5 | 51 | YAY555 | data … |
| 6 | 52 | HAP666 |  |
| 7 | 0 | 000000000 | 00000000000000000000000000000000 |

…

Overflow

|  |  |  |  |
| --- | --- | --- | --- |
| 50 | 54 | GOOD222 | data …. |
| 51 | 0 | XXX555 | data … |
| 52 | 53 | YES666 | data … |
| 53 | 55 | OHNO666 | data … |
| 54 | 0 | LAST222 | data … |
| 55 | 0 | OOPS666 | data … |
| 56 |  |  |  |

lastSynonym with the following iPrimaryRBN values would return the specified RBN:

|  |  |  |
| --- | --- | --- |
| **iPrimaryRBN** | **lastSynonym returns** | **Description** |
| 2 | 54 | Chain is RBN 2, RBN 50, RBN 54 |
| 1 | 0 | The record at RBN 1 is just all 0s |
| 3 | 3 | There is only a primary record |
| 5 | 51 | Chain is RBN 5, RBN 51 |

int **lastSynonym** (FILE \*pFile, int iPrimaryRBN, int iRecSize)

{

Long lRBA;

Record rTemp;

Int iNumRead;

Int rcFseek;

Int iReturnRBN = 0;

For(iRBN = iPrimaryRBN; iRBN!= 0; iRBN = rTemp.iNextChain)

{

lRBA = iRBN \* iRecSize;

rcFseek = fseek(pFile, lRBA, SEEK\_SET);

assert(rcFseek == 0);

iNumRead = fread(&rTemp, iRecSize, 1L, pFile);

if(iNumRead != 1 || rTemp.szKey[0] == ‘\0’)

return iReturnRBN;

iReturnRBN = iRBN;

}

Return iReturnRBN;

}

//Do same routine doing probing

//check if synonym by hashing it

//if return RBN you started with it’s a synonym

//counting loop based on max probes

int **lastSynonym** (FILE \*pFile, int RBN, int iMaxProbes, int iRecSize)

{

Long lRBA;

Record rTemp;

Int iNumRead;

Int rcFseek;

Int iReturnRBN = 0;

For(K=1; K<iMaxProbes; K++)

{

lRBA = RBN+K \* iRecSize;

rcFseek = fseek(pFile, lRBA, SEEK\_SET);

assert(rcFseek == 0);

iNumRead = fread(&rTemp, iRecSize, 1L, pFile);

if(iNumRead != 1 || record.szKey[0] == ‘\0’)

return iReturnRBN;

iReturnRBN = iRBN;

}

Return iReturnRBN;

}