**ENTS 689A – Project 2 Description Karthik Ashok Arjunagi**

**Data Structure:**  Double linked list with a Root node that has the pointers to the head and tail of the list.

1. **Reading and storing the numbers from the file**

**Reading from file:** Read the number in the file character by character and store it in a string. While reading the file, keep track of the number of characters.

**Storing:** The string is created using malloc and a realloc will be done if required.

**Creating blocks:** Take block\_size number of digits at a time from the string and create linked list with these blocks as data. Do this for both the numbers read from the files till all the digits are added to the lists.

**Note:** As we are reading the number from left to right, care should be taken to handle scenarios where the highest digits left don’t complete the block size. This is taken care by checking the remainder of (string\_size/block\_size). If the remainder is non-zero, then (block\_size-remainder) is considered as the first block.

**Example**: number is 56789(String length is 5) and block size is 2.

String\_length%block\_size = 5%2 = 1. Therefore, block\_size-remainder = 2-1 =1.

Hence, block 1 will have 5, block 2 will have 67 and block 3 will have 89.

Also, it is made sure after reading the files that the greater number is considered as the first number and the lesser number is considered as the second number.

1. **Addition**

Add the 2 linked lists node by node (traverse from the tail) and store the sum of blocks in a new linked list. If there is a carry generated by addition of 2 nodes, add this carry to the sum of the next nodes. Each addition operation is considered as 1 block operation and carry is considered as another 1 block operation. Addition with 0 is not counted as a block operation.

Also, if one of the numbers is shorter than the other, then block operations are counted only until all the blocks in the shorter number is added. (Example: num1=12345, num2=67. Block operation is considered only for the addition of 45 and 67. Addition of 23 and 01 with 0 is not considered as block operation).

1. **Subtraction**

Subtraction is also started from the tail of the linked lists and the difference of each node is added to a new linked list. In case of borrow, the next node is subtracted by 1 and the current node is added with 10^block\_size and then the subtraction is done.

Each subtraction is considered as 1 block operation. In case of borrow, the subtraction with 1 of the next node is 1 block operation, addition of 10^block\_size is another block operation and the final subtraction is the last block operation for that block.

Example: 9857

- 76

Borrow: 98-1 = 97 -> 1 block operation (subtraction by 1)

Add borrow: 10^2 + 57 -> 1 block operation (adding 10^block\_size)

Subtract: 157-76 = 81 -> 1 block operation (Subtraction of block 2)

Subtract: 97-0 = 97 -> 0 block operation (Subtraction of block 1)

Final Ans: 9781 -> 3 block operations

1. **Multiplication**

Multiplication is also done from the tail of the linked lists. When 2 blocks are multiplied, the max length of the product can be block\_size\*2. Hence the product can be stored in a string and then read from it and stored in a linked list.

Every product of alternate block is stored in a separate linked list.

Ex: 12345

\* 8967

(00,67)(30,15) -> list1 (67\*45 and 67\*01)

(15, 41) -> list2 (offset from list1 by 1 node from tail) (67\*23)

(00,82)(71,15) -> add list1 and list2 node by node

(00,89)(40, 05) -> list1 (reset the old list1) (89\*45 and 89\*01)

(20, 47) -> list2 (reset the old list2) (89\*23)

(00,01)(10,69)(76,15) -> add the previous sum and the new list1 and list2

The above pattern is continued till all the blocks are considered. The final list will be the product of the 2 numbers. The addition done here can use the addition method created earlier (which returns the number of block operations). The advantage of this method is that we always add 3 lists irrespective of the number of blocks.

Each multiplication of a block is 1 block operation and for addition, it is as mentioned before.

1. **Storing and printing the results**

Each of the result list (addition, subtraction and multiplication) can be read node-by-node and stored in respective strings. These strings can then be stored in the results file and read from there to be displayed.