ICS'12 Spring Homework 11

1) What is a **false fragmentation**? And how to solve this phenomenon? Please list at least two ways and make a simple introduction about them.

2) This problem tests your understanding of memory bugs. Each of the code sequences below may or may not contain memory bugs. The code all compiles without warnings or errors. If you think there is a bug, please write **YES** and indicate the type of bug from the list below of memory bugs. Otherwise, if you think there are no memory bugs in the code, please write **NO**.

   Bugs:
   1. buffer overflow error
   2. memory leak
   3. potential for dereferencing a bad pointer
   4. incorrect use of free

A:

```
/*
* An attempt to write a safe version of strndup() function
* Note: For this problem, assume that if the function returns a
* non-NULL pointer to dest, then the caller eventually frees the dest buffer.
*/

char *strndup(char *src, int max)
{
      char *dest;
      int i;

      if (!src || max <= 0)
          return NULL;

      dest = (char*)malloc(max);
      for (i=0; i < max && src[i] != '\0'; i++)
      {
```

```
            dest[i] = src[i];
        }
        dest[i] = '\0';
        return dest;
         }
```

_____Type of bug:

B:

```
/*
* Note: For this problem, the function below reverses a single linked list, and
* returns the head node of the new list.
*/

typedef int ElemType;
typedef struct Node
{
       ElemType data;
       struct Node *next;
}Node,*LinkList;

LinkList ListReverse(LinkList L)
{
      Node *temp;
      temp = L;
      LinkList nList;
     nList = (LinkList)malloc(sizeof(LinkList))

      nList->data = L->data;
      nList->next = NULL;

      while(L->next != NULL)
      {
            temp = nList->next;
            nList->next = L->next;
            L->next = L->next->next;
            nList->next->next = temp;

      }
      free(L);
      return nList;

}
```

_____Type of bug:

3) Show the type of locality (temporal or spatial) for each object.

```
Code:
int a[10][20];
int b[10];
for (int i = 0; i < 10; ++i)
{
    for (int j = 0; j < 20; ++j)
    {
        a[i][j] += b[i];
    }
}
```

Object	Type of Locality
a	
b	
i	
j	

4) Do you think placement strategy and replacement strategy affect the miss rate of a cache? Give your reasons.