Problem HT-3 (2 parts)

Hash Table Implementation

```c
void Insert(HashTable *HT, int Key, int Value) {
    Link *FoundLink;

    FoundLink = Find_Key(HT, Key);
    if (FoundLink != NULL)
        FoundLink->Value = Value;
    else {
        if (HT->FreeLinks == NULL)
            HT->FreeLinks = Make_New_Links();
        FoundLink = HT->FreeLinks;
        HT->FreeLinks = HT->FreeLinks->Next;
        FoundLink->Key = Key;
        FoundLink->Value = Value;
        FoundLink->Next = HT->Buckets[Hash(HT, Key)];
        HT->Buckets[Hash(HT, Key)] = FoundLink;
        HT->Size += 1;
        if (HT->Size > HT->NumBuckets * RESIZERATIO)
            Resize_Hash_Table(HT);
    }
}
```

Part A: Complete the C function Find_Key that searches the hash table for an entry corresponding to a specified key. It should return a pointer to the matching Link entry if the key is found or return NULL if the key is not found in the hash table.

```c
Link *Find_Key(HashTable *HT, int Key) {
    Link *ThisLink;
    int Index;
    int Hash = Hash(HashTable *HT, int Key);

    ThisLink = HT->Buckets[Hash];
    while (ThisLink != NULL && ThisLink->Key != Key) {
        ThisLink = ThisLink->Next;
    }

    return ThisLink;
}
```

Part B: The following questions are related to the hash table implementation listed above.

When Insert is called more than once with the same key, what occurs?
Where is a new entry link placed on the bucket list?
What is the range of possible values for average bucket list length?
Describe the type of Buckets in HashTable (in 10 words or less).
What is the size (in bytes) of HashTable?
What is the size (in bytes) of the initial Buckets array?