資料結構 HW5 report 徐松廷 110501521

1. 目標:延續先前作業的程式，增加支援類 Redis Sorted Set 的操作，指令包含 ZADD, ZCARD, ZCOUNT, ZINTERSTORE, ZUNIONSTORE, ZRANGE, ZRANGEBYSCORE, ZRANK, ZREM, ZREMRGEBYSCORE
2. Code review:

   這次的 task 是使用 sorted set 來對 Redis 當中的 key-value pair 進行排序，排序的方式是依照 value 的大小，所以 value 的型態只能是可比較的型態，這邊我們按照官方定義將 value 型態統一設定成 int。以下說明 ZADD, ZCARD, ZCOUNT, ZINTERSTORE, ZUNIONSTORE, ZRANGE, ZRANGEBYSCORE, ZRANK, ZREM, ZREMRGEBYSCORE 共十種指令的實現方式:

   a. Node 結構與 sorted set 的定義:基本上就是先定義好一個 node 後再用 linked list 的方式把 node 串接起來，這邊用 Linked list 而非 array 的原因是不確定資料個數。

```c
typedef struct SS_Node {
    int data;
    char* member; // Node name
    struct SS_Node* next;
}SS_Node;

// Define a structure for the sorted set
typedef struct SortedSet {
    char name[50];
    SS_Node* head;
} SortedSet;
```

   b. 建立並初始化 sorted set:這邊應該沒甚麼問題，就是動態配置出記憶體空間，並定義一個建立 node 的 function 方便之後使用。

```c
SortedSet* create_sorted_set(char* set_name){
    SortedSet* sorted_set=(SortedSet*)malloc(sizeof(SortedSet));
    sorted_set->head=NULL;
    strcpy(sorted_set->name, set_name);
    return sorted_set;
}
```

```c
SS_Node* create_Node(int data, char* member) {
    SS_Node* newNode = (SS_Node*)malloc(sizeof(SS_Node));
    if (newNode == NULL) {
        printf("Memory allocation failed!\n");
        exit(1);
    }
    newNode->data = data;
    newNode->member = strdup(member);
    newNode->next = NULL;
    return newNode;
}
```

   c. Set 的排序 function:sort_set()是一個負責進行 set 排序的 function，我們這邊是用 merge sort 實現。

```c
SS_Node* mergeSort(SS_Node* head) {
    if (head == NULL || head->next == NULL) {
        return head;
    }
    SS_Node* middle = head;
    SS_Node* end = head->next;
    while (end != NULL && end->next != NULL) {
        middle = middle->next;
        end = end->next->next;
    }
    SS_Node* secondHalf = middle->next;
    middle->next = NULL;
    return merge(mergeSort(head), mergeSort(secondHalf));
}
void sortSet(SortedSet* set) {
    set->head = mergeSort(set->head);
}
```

```c
SS_Node* merge(SS_Node* list1, SS_Node* list2) {
    if (list1 == NULL) {
        return list2;
    }
    if (list2 == NULL) {
        return list1;
    }
    SS_Node* result = NULL;
    if (list1->data <= list2->data) {
        result = list1;
        result->next = merge(list1->next, list2);
    } else {
        result = list2;
        result->next = merge(list1, list2->next);
    }

    return result;
}
```

d. ZADD 功能: insertElement()負責執行 ZADD 指令，傳入一個 node 的 member(也就是 key)和 data(就是 value)，建立完成之後會進行 set 的重新排序。

```c
void insertElement(SortedSet* set, int data, char* member) {
    SS_Node* head=set->head;
    int over_write_bool=0;
    while(head){
        if(strcmp(head->member,member)==0){
            //覆蓋掉之前的data
            head->data=data;
            over_write_bool=1;
        }
        head=head->next;
    }
    if(over_write_bool==0){
        SS_Node* newNode = create_Node(data, member);
        newNode->next = set->head;
        set->head = newNode;
        sortSet(set);
    }
}
```

e. ZREM:此功能用 deleteMemberNodes()實現，傳入要刪除的 member(也就是 key)，會用 call by address 的方式修改到 sorted set 並刪除該 node.

```c
void deleteMemberNodes(SortedSet* set, char* member) {
    SS_Node* current = set->head;
    SS_Node* prev = NULL;
    while (current != NULL) {
        if (strcmp(current->member, member) == 0) {
            if (prev == NULL) {
                SS_Node* temp = current;
                set->head = current->next;
                current = current->next;
                free(temp);
            } else {
                prev->next = current->next;
                free(current);
                current = prev->next;
            }
        } else {
            // Move to the next node
            prev = current;
            current = current->next;
        }
    }
}
```

f. ZRANGE: 實現 ZRANGE 功能時，我們會將 set 的起始 index 和中指 index 傳入 displaySet()，例如 ZRANGE set1 0 4 就是輸出 set1 的第 0 組到第 4 組 key-value pairs。

```c
void displaySet(SortedSet* set, int start,int end) {

    printf("Sorted Set %s: ", set->name);
    SS_Node* current = set->head;
    int index=0;
    while (current != NULL) {
        if(index>=start&&(index<=end||end==-1)){
            printf("(%s: %d) ", current->member, current->data);
            index++;
        }
        current = current->next;
    }
    printf("\n");
}
```

g. ZCOUNT:回傳該 set 有幾組 key-value pairs，直接輸出數量。

```
void ZCOUNT(SortedSet* set,int low_b,int up_b){
    SS_Node* cur = set->head;
    int index=0;
    while (cur != NULL) {
        if(cur->data>=low_b&&cur->data<=up_b){
            index++;
        }
        cur=cur->next;
    }
    printf("%d\n",index);
```

h. ZRANK:這個功能是輸入一個 key(字串)，會回傳該組 key-value pair 的 Index 在 sorted set 中的順位，我們一樣是用查找的方式實現。

```
void ZRANK(SortedSet* set,char* member){
    int rank=0;
    int found_bool=0;
    SS_Node* cur=set->head;
    while(cur){
        if(strcmp(cur->member,member)==0){
            found_bool=1;
            break;
        }
        rank++;
        cur=cur->next;
    }
    if(found_bool==0)printf("%s not found in %s\n",member,set->name);
    else printf("%d\n",rank);
}
```

i. ZRANGEBYSCORE: 這個功能是使用者必須輸入 score(也就是 value)的上限和下限，本函數會將在範圍內的 key-value pair 按照順序輸出，最小下限是負無窮，最大上限是正無窮。

```
void ZRANGEBYSCORE(SortedSet* set,char *low_b,char* up_b){
    int max_int = INT_MAX;
    int min_int = INT_MIN;
    if(strcmp(low_b,"-inf")==0)min_int = INT_MIN;
    else min_int = atoi(low_b);
    if(strcmp(up_b,"+inf")==0)max_int = INT_MAX;
    else max_int = atoi(up_b);

    SS_Node* cur=set->head;
    while(cur){
        if(cur->data>=min_int&&cur->data<=max_int){
            printf("(%s: %d) ", cur->member, cur->data);
        }
        cur=cur->next;
    }
    printf("\n");
}
```

j. ZREMRANGEBYSCORE: 這個功能是使用者必須輸入 score(也就是 value)的上限和下限，本函數會將 socore(value)大於下限小於上限的 key-value pair 刪除，最小下限是負無窮，最大上限是正無窮。

```
void ZREMRANGEBYSCORE(SortedSet* set,char *low_b,char* up_b){
    int max_int = INT_MAX;
    int min_int = INT_MIN;
    if(strcmp(low_b,"-inf")==0)min_int = INT_MIN;
    else min_int = atoi(low_b);
    if(strcmp(up_b,"+inf")==0)max_int = INT_MAX;
    else max_int = atoi(up_b);
    char* member_to_remove[100];
    int member_to_remove_ct=0;
    SS_Node* cur=set->head;
    while(cur){
        if(cur->data>=min_int&&cur->data<=max_int){
            member_to_remove[member_to_remove_ct]=cur->member;
            member_to_remove_ct++;
        }
        cur=cur->next;
    }
    for(int i=0;i<member_to_remove_ct;i++){
        deleteMemberNodes(set, member_to_remove[i]);
    }
```

k. ZUNIONSTORE:本次的目標是傳入數個 sorted set 後取聯集，再回傳該 sorted set，我這邊是先取兩個 set 的交集，排序好後再回傳，實際在使用時會重複呼叫此 function 直到取完所有 set 的聯集。當兩個 set 有相同的 key 時，我是根據官方定義的方式將兩者 value 相加後合併成單一一個 node，具體實現方式如下:

```
SortedSet* find_union_sorted_set(SortedSet* set1,SortedSet* set2,char* union_name){
    SortedSet* union_set=create_sorted_set(union_name);
    SS_Node* set1_head=set1->head;
    SS_Node* set2_head=set2->head;
    while(set1_head){
        insert_without_sorting(union_set,set1_head->data,set1_head->member);
        set1_head=set1_head->next;
    }
    while(set2_head){
        set1_head=set1->head;
        int collision_bool=0;
        while(set1_head){
            if(strcmp(set1_head->member,set2_head->member)==0){
                deleteMemberNodes(union_set,set1_head->member);
                insert_without_sorting(union_set,set1_head->data+set2_head->data,set1_head->member);
                collision_bool=1;
            }
            set1_head=set1_head->next;
        }
        if(collision_bool==0){
            insert_without_sorting(union_set,set2_head->data,set2_head->member);
        }
        set2_head=set2_head->next;
    }
    sortSet(union_set);
    return union_set;
```

l. ZINTERSTORE:本次的目標是傳入數個 sorted set 後取交集，再回傳該 sorted set，我這邊是先取兩個 set 的交集，排序好後再回傳，實際在使用時會重複呼叫此 function 直到取完所有 set 的交集。當兩個 set 有相同的 key 時，我是根據官方定義的方式將兩者 value 相加後合併成單一一個 node，具體實現方式如下:

```
SortedSet* find_intersection_set(SortedSet* set1,SortedSet* set2,char* intersected_name){
    SortedSet* intersected_set=create_sorted_set(intersected_name);
    SS_Node* set1_head=set1->head;
    SS_Node* set2_head=set2->head;

    while(set2_head){
        set1_head=set1->head;
        while(set1_head){
            if(strcmp(set1_head->member,set2_head->member)==0){
                insert_without_sorting(intersected_set,set1_head->data+set2_head->data,set1_head->member);
            }
            set1_head=set1_head->next;
        }
        set2_head=set2_head->next;
    }
    sortSet(intersected_set);
    return intersected_set;
}
```

3. 測試與執行結果

編譯方式:

gcc -c redis_dll.c -o redis_dll.o

gcc -c redis_str.c -o redis_str.o

gcc -o main main.c redis_dll.o redis_str.o

Linux Ubuntu 執行結果如下:

```
eason@LAPTOP-Q69P3FAE:/mnt/c/Users/user/DS_HW5$ gcc -o main main.c
eason@LAPTOP-Q69P3FAE:/mnt/c/Users/user/DS_HW5$ ./main
Please input the command (ZADD,ZCARD,ZCOUNT,ZINTERSTORE,ZUNIONSTORE,ZRANGE,ZRANGEBYSCORE,ZRANK,ZREM,ZREMRANG
E,ZREMRGEBYSCORE)
ZADD set1 1 A 2 B 30 C 5 D 67 E 12 F 134 G 98 H

Please input the command (ZADD,ZCARD,ZCOUNT,ZINTERSTORE,ZUNIONSTORE,ZRANGE,ZRANGEBYSCORE,ZRANK,ZREM,ZREMRANG
E,ZREMRGEBYSCORE)
ZADD set2 199 X 21 N 30 D 511 E 671 F 981 L

Please input the command (ZADD,ZCARD,ZCOUNT,ZINTERSTORE,ZUNIONSTORE,ZRANGE,ZRANGEBYSCORE,ZRANK,ZREM,ZREMRANG
E,ZREMRGEBYSCORE)
ZADD set3 314 D 21 E 30 J 123 T 61 Q 82 Y 987 C
```

依序用 ZADD 建立 set1 set2 set3

```
Please input the command (ZADD,ZCARD,ZCOUNT,ZINTERSTORE,ZUNIONSTORE,ZRANGE,ZRANGEBYSCORE,ZRANK,ZREM,ZREMRANG
E,ZREMRGEBYSCORE)
ZRANGE set1 0 -1
Sorted Set set1: (A: 1) (B: 2) (D: 5) (F: 12) (C: 30) (E: 67) (H: 98) (G: 134)

Please input the command (ZADD,ZCARD,ZCOUNT,ZINTERSTORE,ZUNIONS
E,ZREMRGEBYSCORE)
ZREM set1 A B

Please input the command (ZADD,ZCARD,ZCOUNT,ZINTERSTORE,ZUNIONSTORE,ZRANGE,ZRANGEBYSCORE,ZRANK,ZREM,ZREMRANG
E,ZREMRGEBYSCORE)
ZRANGE set1 0 -1
Sorted Set set1: (D: 5) (F: 12) (C: 30) (E: 67) (H: 98) (G: 134)

Please input the command (ZADD,ZCARD,ZCOUNT,ZINTERSTORE,ZUNIONSTO
E,ZREMRGEBYSCORE)
ZRANGE set3 0 -1
Sorted Set set3: (E: 21) (J: 30) (Q: 61) (Y: 82) (T: 123) (D: 314) (C: 987)
```

ZRANGE set1 0 -1 會列出所有 set1 pairs

ZREM set1 A B 會移除 key 為 A、B 的 pairs

A B 被移除後 set1 的內容

ZRANGE set3 0 -1 會列出所有 set3 pairs

```
Please input the command (ZADD,ZCARD,ZCOUNT,ZINTERSTORE,ZUNIONSTORE,ZRANGE,ZRANGEBYSCORE,ZRANK,ZREM,ZREMRANG
E,ZREMRGEBYSCORE)
ZRANGE set1 0 4
Sorted Set set1: (D: 5) (F: 12) (C: 30) (E: 67) (H: 98)

Please input the command (ZADD,ZCARD,ZCOUNT,ZINTERSTORE,ZUNIONSTORE,ZRANGE,ZRANGEBYSCORE,ZRANK,ZREM,ZREMRANG
E,ZREMRGEBYSCORE)
ZRANK set1 H
4

Please input the command (ZADD,ZCARD,ZCOUNT,ZINTERSTORE,ZUNIONSTORE,ZRANGE,ZRANGEBYSCORE,ZRANK,ZREM,ZREMRANG
E,ZREMRGEBYSCORE)
ZCARD set1
6

Please input the command (ZADD,ZCARD,ZCOUNT,ZINTERSTORE,ZUNIONSTORE,ZRANGE,ZRANGEBYSCORE,ZRANK,ZREM,ZREMRANG
E,ZREMRGEBYSCORE)
ZRANGEBYSCORE set1 -inf 90
(D: 5) (F: 12) (C: 30) (E: 67)
```

ZRANGE set1 0 4 會列出 set1 第 0 個到第 4 個 member

ZRANK set1 H 會列出 set1 中 member 為 H 的順位

ZCARD set1 會列出 set1 的 pair 數量

ZRANGEBYSCORE set1 -inf 90 會列出 value 大於-inf 小於 90 的 pair

```
Please input the command (ZADD,ZCARD,ZCOUNT,ZINTERSTORE,ZUNIONSTORE,ZRANGE,ZRANGEBYSCORE,ZRANK,ZREM,ZREMRANG
E,ZREMRGEBYSCORE)
ZRANGEBYSCORE set1 20 +inf
(C: 30) (E: 67) (H: 98) (G: 134)

Please input the command (ZADD,ZCARD,ZCOUNT,ZINTERSTORE,ZUNIONSTORE,ZRANGE,ZRANGEBYSCORE,ZRANK,ZREM,ZREMRANG
E,ZREMRGEBYSCORE)
ZRANGEBYSCORE set1 20 90
(C: 30) (E: 67)

Please input the command (ZADD,ZCARD,ZCOUNT,ZINTERSTO
E,ZREMRGEBYSCORE)
ZINTERSTORE inter_set 3 set1 set2 set3
```

ZRANGEBYSCORE set1 20 +inf 會列出 value 大於 20 小於+inf 的 pair

ZRANGEBYSCORE set1 20 90 會列出 value 大於 20 小於 90 的 pair

ZINTERSTORE inter_set 3 set1 set2 set3 表示將三個 set (set1,set2,set_3)取交集後建立成 inter_set

```
Please input the command (ZADD,ZCARD,ZCOUNT,ZINTERSTORE,ZUNIONSTORE,ZRANGE,ZRANGEBYSCORE,ZRANK,ZREM,ZREMRANG
E,ZREMRGEBYSCORE)
ZRANGE inter_set 0 -1
Sorted Set inter_set: (D: 349) (E: 599) (F: 4139)

Please input the command (ZADD,ZCARD,ZCOUNT,ZINTERSTORE,ZUNIONSTORE,ZRANGE,ZRANGEBYSCORE,ZRANK,ZREM,ZREMRANG
E,ZREMRGEBYSCORE)
ZUNIONSTORE uni_set 3 set1 set2 set3
```

使用 ZRANGE 查看 inter_set 內容:確認取了交集(相同 key 的 value 會相加)，功能正確。

ZUNIONSTORE uni_set 3 set1 set2 set3 表示將三個 set (set1,set2,set_3)取聯集後建立成 uni_set.

使用 ZRANGE 查看 uni_set 內容: 確認取了聯集(相同 key 的 value 會相加)，功能正確。

```
Please input the command (ZADD,ZCARD,ZCOUNT,ZINTERSTORE,ZUNIO
E,ZREMRGEBYSCORE)
ZRANGE uni_set 0 -1
Sorted Set uni_set: (N: 21) (J: 30) (OI: 34) (Q: 61) (WE: 78) (VR: 78) (Y: 82) (H: 98) (T: 123) (G: 134) (X:
 199) (D: 349) (E: 599) (L: 981) (C: 1017) (F: 4139)

Please input the command (ZADD,ZCARD,ZCOUNT,ZINTERSTORE,ZU
E,ZREMRGEBYSCORE)
ZCOUNT set3 25 250
5
```

再次測試 ZCOUNT 功能，以 set3 而言有 J, Q, VR, Y, T 總共 5 個 pairs 在範圍內，故功能正確

```
Please input the command (ZADD,ZCARD,ZCOUNT,ZINTERSTORE,ZUNIONSTORE,ZRANGE,ZRANGEBYSCORE,ZRANK,ZREM,ZREMRANG
E,ZREMRGEBYSCORE)
ZRANGE set3 0 -1
Sorted Set set3: (E: 21) (J: 30) (Q: 61) (VR: 78) (Y: 82) (T: 123) (D: 314) (C: 987) (F: 3456)

Please input the command (ZADD,ZCARD,ZCOUNT,ZINTERSTORE,ZUNIONSTORE,ZRANGE,ZRANGEBYSCORE,ZRANK,ZREM,ZREMRANG
E,ZREMRGEBYSCORE)
ZREMRANGEBYSCORE set1 -inf 60

Please input the command (ZADD,ZCARD,ZCOUNT,ZINT
E,ZREMRGEBYSCORE)
ZRANGE set1 0 -1
Sorted Set set1: (E: 67) (H: 98) (G: 134)
```

再次測試 ZREMRANGEBYSCORE，原本的 set1 有(5, 12, 30 ,67, 98, 134), ZREMRANGEBYSCORE set1 -inf 60 將小於 60 的 pair 移除，只剩 67, 98, 134 ，功能正確

4. 說明

基本上這次的作業我覺得最難的部分是對一個 Linked list 的結構去做 merge sort(基本上也是不斷切割，切割到不能再切之後，就開始合併，合併的方式也是兩個小 linked list 比頭的大小，一步一步串起來)。剩下我覺得就是 C 語言的字串讀取需要畫花心思刻出來，畢竟這次要支援十種語法，而且語法的結構除了開頭必定是 Command，之後是 set_name 之外，後面的結構都不太一樣，有時要讀取字串後轉成數字，有時則是字串判斷，這邊蠻需要花時間處理。

感謝助教批改!!!