```c
#include <stdio.h>
#include <stdlib.h>

struct floor
{
    float Area ;
    int similar;
    int classOfApp;
    float totKw;
    struct floor *next ;
};

struct floor *append(struct floor *, struct floor *);   //fn to append the linked list
struct floor *allSim();  // return the start of the linked list
struct floor *createPlace(int);   // allocate a place of the digits given
void loadEstimation(struct floor *); // calculate the load of the building
void freeMemory(struct floor *);  //free the memory
void printRes(struct floor *, int); //print the linked list
void searchCB(int KVA);


int main()
{
    //!showMemory (start = 65520)
    //!showMemory (start = 272)
    struct floor *start;  // take the start of linked list
    int nOfFloors; // take the no of floors
    printf("Please Enter the Number of the floors at the building : \n");
    scanf("%d", &nOfFloors);
    start = allSim();
    loadEstimation(start);
    printRes(start, nOfFloors);
    freeMemory(start);
    return 0 ;
}


struct floor *allSim()
{
    /*start to hold the start of linked list
    end to hold the end of the current
    new to hold the value of the next location*/
    struct floor *start, *end, *newptr;

    int nOfapp;//hold the number of appartments at the floor
```

```c
    int sim = 0 ; //hold the number of the similar if there are
    int i;//counter for the appartments
    int newNo; // hold the new no of appartments after subtract the similar

    printf("Enter the number of the appartments at the floor: \n");
    scanf("%d", &nOfapp);
    printf("Enter if there is similar appartments at floor: \n");
    scanf("%d", &sim);
    start = NULL;
    newNo = nOfapp - sim;

    for (i = 0; i <= newNo; i++)
    {

        if (sim == 0) // check if similar equals zero so that i have to take all of them individually
        {
            while (newNo != 0)
            {
                newptr = createPlace(newNo);
                if (start == NULL)
                {
                    start = newptr;
                    end = start;
                }
                else
                {
                    end = append(end, newptr);

                }
                newNo--;

            }

        }
        else  //i have to check the number of similar then sub them from the no of appartments
        {
            newptr = createPlace(sim);// allocate a place for those similar locations

            if (start == NULL) // append the linked list
            {
                start = newptr;
                end = start;
            }
            else
            {
```

```c
                end = append(end, newptr);

            }

            if (newNo > 1) // check if newNo > 1 so that there can be similar appartments
            {

                printf("Enter if there is another similar "
                "appartments in the remaining: %d appartments.\n", newNo);
                scanf("%d", &sim);
                if (newNo - sim != 0)
                {
                    newNo = newNo - sim;
                }
                else // similar = the number of the remaining appartments
                {
                    i = 0; //set the counter to equal zero
                    sim = newNo; // return that the similar equals to the number of remaining appartments
                    newNo = 0;//say that there is no remaining appartments
                    i--;// sub 1 from counter not to go out from the for loop

                }



            }
            else // if the newNumber equals 1
            {
                sim = 0;
                i = 0;
            }

        }


    }

    return start ;

}

struct floor *createPlace(int sim)
{
    struct floor *ptr ;
    ptr = (struct floor *) malloc(sizeof(struct floor));
```

```c
    ptr->similar = sim;
    if (ptr->similar > 1)
    {
        printf("Enter the area of an appartment of these  %d appartments : \n", ptr->similar);
        scanf("%f", &(ptr->Area));
        //printf("Enter the class of an appartment of these  %d appartments : \n", ptr->similar);
        //scanf("%d", &(ptr->classOfApp));
    }
    else
    {
        printf("Enter the area of the remaining appartment : \n");
        scanf("%f", &(ptr->Area));
        //printf("Enter the class of the remaining appartment : \n");
        //scanf("%d", &(ptr->classOfApp));
    }
    ptr->next = NULL;
    return ptr ;
}

struct floor *append(struct floor *end, struct floor *newptr)
{
    end->next = newptr;
    return (end->next);

}
void loadEstimation(struct floor *start)
{
    struct floor *ptr ;
    float maxLighingD, maxSocketsD, maxAcD;
    ptr = start ;
    while (ptr != NULL)
    {
        //lighting loads
        maxLighingD = ptr->Area * 15 * .6;
        //Sockets loads
        maxSocketsD = ptr->Area * 40 * .7;
        //Air conditioning loads
        maxAcD = ptr->Area * 65 * .7;
        ptr->totKw = ptr->similar * (maxLighingD + maxSocketsD + maxAcD);
        ptr = ptr->next;
    }
}

void freeMemory(struct floor *start)
{
```

```c
    struct floor *ptr ;
    ptr = start ;
    while (ptr != NULL)
    {
        start = ptr;
        ptr = ptr->next;
        free(start);
    }
}
void printRes(struct floor *start, int nOfFloors)
{
    struct floor *ptr ;
    ptr = start ;
    float totOfAll = 0;
    float totKVA ;
    while (ptr != NULL)
    {
        if ((ptr->similar) > 1)
        {
            printf("\n");
            printf("The Total KW of the %d similar "
            "appartments equals : %.2f KW.\n", ptr->similar, (ptr->totKw) / 1000);

            printf("The required CB of Each appartment is :\n");
            searchCB((ptr->totKw) / ((ptr->similar)*1000));
        }
        else
        {
            printf("\n");
            printf("The KW of the remaining appartment equals : %.2f KW\n", (ptr->totKw) / 1000);
            printf("The required CB of them is  :\n");
            searchCB((ptr->totKw) / ((ptr->similar)*1000));
        }
        totOfAll = totOfAll + ptr->totKw;
        ptr = ptr->next;
    }
    totKVA = (nOfFloors * totOfAll) / 1000 ;
    printf("\n");
    printf("The Total KVA of the entire building equals : %.2f KW. \n", totKVA);
    if(totKVA <= 648)
    {
    printf("The required CB of the whole building is :\n");
    searchCB(totKVA);
    printf("\n");
    }
```

```c
    else
    {
        printf("\n");
        printf("Because the total KVA is very large (NO single CB Can carry it) you will need to divide it into "
        "two CB's or more :\n");
        printf("Use 3 CB's : \n");
        searchCB(totKVA / 3);
        printf("\n");
    }
}

void searchCB(int KVA)
{

    FILE *fpointer;
    int id , i, y, CB;
    char array[255];
    CB = KVA ;

    fpointer = fopen("cb.txt", "r");

    for (i = 0; i < 100; i++)
    {
        if (CB >= 4 && CB <= 16)
        {
           for (y = 0; y < 100; y++)
           {
                fscanf(fpointer, "%s", array);
                id = atoi(array);
                if(id == 4)
                {
                    fgets(array, 255,(FILE *)fpointer);
                    printf("%s",array);
                    break;
                }

            }
            break;
        }
        else if (CB > 16 && CB <= 24)
        {
            for (y = 0; y < 100; y++)
            {
                fscanf(fpointer, "%s", array);
                id = atoi(array);
```

```
271                 if(id == 17)
272                             {
273                     fgets(array,255, (FILE *)fpointer);
274                     printf("%s", array);
275                     break;
276                 }
277             }
278         break;
279         }
280         else if (CB > 24 && CB <= 32)
281         {
282             for (y = 0; y < 100; y++)
283             {
284                 fscanf(fpointer, "%s", array);
285                 id = atoi(array);
286                 if(id == 25)
287                             {
288                     fgets(array, 255, (FILE *)fpointer);
289                     printf("%s", array);
290                     break;
291                 }
292 
293             }
294             break;
295         }
296                 else if (CB > 32 && CB <= 40)
297                 {
298             for (y = 0; y < 100; y++)
299             {
300                 fscanf(fpointer, "%s", array);
301                 id = atoi(array);
302                 if(id == 33)
303                             {
304                     fgets(array, 255, (FILE *)fpointer);
305                     printf("%s", array);
306                     break;
307                 }
308             }
309             break;
310         }
311                 else if (CB > 40 && CB <= 56)
312                 {
313             for (y = 0; y < 100; y++)
314             {
315                 fscanf(fpointer, "%s", array);
```

```
316                 id = atoi(array);
317                 if(id == 41)
318                 {
319                     fgets(array, 255, (FILE *)fpointer);
320                     printf("%s", array);
321                     break;
322                 }
323             }
324             break;
325         }
326                 else if (CB > 56 && CB <= 83)
327                 {
328             for (y = 0; y < 100; y++)
329             {
330                 fscanf(fpointer, "%s", array);
331                 id = atoi(array);
332                 if(id == 57)
333                             {
334                     fgets(array, 255, (FILE *)fpointer);
335                     printf("%s", array);
336                     break;
337                 }
338             }
339             break;
340         }
341                 else if (CB > 83 && CB <= 103)
342                 {
343             for (y = 0; y < 100; y++)
344             {
345                 fscanf(fpointer, "%s", array);
346                 id = atoi(array);
347                 if(id == 84)
348                             {
349                     fgets(array, 255, (FILE *)fpointer);
350                     printf("%s", array);
351                     break;
352                 }
353             }
354             break;
355         }
356                 else if (CB > 103 && CB <= 126)
357                 {
358             for (y = 0; y < 100; y++)
359             {
360                 fscanf(fpointer, "%s", array);
```

```
361                 id = atoi(array);
362                 if(id == 104)
363                             {
364                     fgets(array, 255, (FILE *)fpointer);
365                     printf("%s", array);
366                     break;
367                 }
368 
369             }
370             break;
371         }
372                 else if (CB > 127 && CB <= 166)
373                 {
374             for (y = 0; y < 100; y++)
375             {
376                 fscanf(fpointer, "%s", array);
377                 id = atoi(array);
378                 if(id == 128)
379                             {
380                     fgets(array, 255, (FILE *)fpointer);
381                     printf("%s", array);
382                 break;
383                 }
384             }
385             break;
386         }
387                 else if (CB > 166 && CB <= 206)
388                 {
389             for (y = 0; y < 100; y++)
390             {
391                 fscanf(fpointer, "%s", array);
392                 id = atoi(array);
393                 if(id == 167)
394                             {
395                     fgets(array, 255, (FILE *)fpointer);
396                     printf("%s", array);
397                     break;
398                 }
399             }
400             break;
401         }
402                 else if (CB > 206 && CB <= 246)
403                 {
404             for (y = 0; y < 100; y++)
405             {
```

```
406                 fscanf(fpointer, "%s", array);
407                 id = atoi(array);
408                 if(id == 207)
409                             {
410                     fgets(array, 255, (FILE *)fpointer);
411                     printf("%s", array);
412                     break;
413                 }
414             }
415             break;
416         }
417                 else if (CB > 246 && CB <= 326)
418                 {
419             for (y = 0; y < 100; y++)
420             {
421                 fscanf(fpointer, "%s", array);
422                 id = atoi(array);
423                 if(id == 247)
424                             {
425                     fgets(array, 255, (FILE *)fpointer);
426                     printf("%s", array);
427                     break;
428                 }
429 
430             }
431             break;
432         }
433                 else if (CB > 326 && CB <= 406)
434                 {
435             for (y = 0; y < 100; y++)
436             {
437                 fscanf(fpointer, "%s", array);
438                 id = atoi(array);
439                 if(id == 327)
440                             {
441                     fgets(array, 255, (FILE *)fpointer);
442                     printf("%s", array);
443                     break;
444                 }
445             }
446             break;
447         }
448                 else if (CB > 406 && CB <= 486)
449                 {
450             for (y = 0; y < 100; y++)
```

```
451                 {
452                     fscanf(fpointer, "%s", array);
453                     id = atoi(array);
454                     if (id == 407)
455                                 {
456                         fgets(array, 255, (FILE*)fpointer);
457                         printf("%s", array);
458                         break;
459                     }
460                 }
461                 break;
462             }
463                     else if (CB > 486 && CB <= 648)
464                     {
465                 for (y = 0; y < 100; y++)
466                 {
467                     fscanf(fpointer, "%s", array);
468                     id = atoi(array);
469                     if (id == 487)
470                                 {
471                         fgets(array, 255, (FILE *)fpointer);
472                         printf("%s", array);
473                         break;
474                     }
475                 }
476                 break;
477             }
478         }
479 
480 
481         fclose(fpointer);
482         return ;
483     }
```