TI Projekt

https://www.geeksforgeeks.org/function-pointer-in-c/

https://www.tutorialspoint.com/c_standard_library/c_function_qsort

Live Demo

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

int values[] = { 88, 56, 100, 2, 25 };

int cmpfunc (const void * a, const void * b) {
   return ( *(int*)a - *(int*)b );
}

int main () {
   int n;

   printf("Before sorting the list is: \n");
   for( n = 0 ; n < 5; n++ ) {
      printf("%d ", values[n]);
   }

   qsort(values, 5, sizeof(int), cmpfunc);

   printf("\nAfter sorting the list is: \n");
   for( n = 0 ; n < 5; n++ ) {
      printf("%d ", values[n]);
   }

   return(0);
}
```

Cmpfunc macht sich das leben leicht es returnt a-b wenn die beiden gleich sind dann wird es null wenn a größer ist kommt eine positive zahl wenn b größer eine negative zahl

Strcomp → library funktion

String.h ist das header file für string compare

```c
qsort(values, 5, sizeof(int), cmpfunc);
```

Values → die werte

5 → anzahl der elemente

Sizeof(int) → größe eines elements

Cmpfunc → übergabe der comparefunktion

https://www.tutorialspoint.com/c_standard_library/c_function_bsearch

Live Demo

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


int cmpfunc(const void * a, const void * b) {
   return ( *(int*)a - *(int*)b );
}

int values[] = { 5, 20, 29, 32, 63 };

int main () {
   int *item;
   int key = 32;

   /* using bsearch() to find value 32 in the array */
   item = (int*) bsearch (&key, values, 5, sizeof (int), cmpfunc);
   if( item != NULL ) {
      printf("Found item = %d\n", *item);
   } else {
      printf("Item = %d could not be found\n", *item);
   }

   return(0);
}
```

Für bsearch brauche ich ein bereits sortiertes array

Listen Verkettung ändern:

In welcher reinfolge müssen dir Zeiger umgehängt werden.