```c
//
// Created by hfwei on 2023/12/13.
// Question: What if char key_name[] = "Zhang Chu"?
//

#include <stdio.h>
#include <string.h>
#include <stdbool.h>

// See https://codebrowser.dev/glibc/glibc/stdlib/stdlib.h.html#__compar_fn_t
// The first is a pointer to the key for the search,
// and the second is a pointer to the array element to be compared with the key.
typedef int (*__compar_fn_t)(const void *, const void *);

// See https://codebrowser.dev/glibc/glibc/bits/stdlib-bsearch.h.html#19
void *bsearch(const void *__key, const void *__base, size_t __nmemb, size_t __size,
              __compar_fn_t __compar);

int CompareStrs(const void *left, const void *right);
int CompareStrsCI(const void *left, const void *right);
int CompareStrsAddress(const char *left, const char *right);

// int (*GetCompareFunction(bool case_sensitive))(const void *, const void *);
__compar_fn_t GetCompareFunction(bool case_sensitive) {
  return case_sensitive ? &CompareStrs : &CompareStrsCI;
}

const char *names[] = {
    "Cui Jian",
    "Dou Wei",
    "ErShou Rose",
    "Hu Mage",
    "Li Zhi",
    "Luo Dayou",
    "Wan Qing",
    "Yao",
    "Zhang Chu",
    "ZuoXiao",
};

int main(void) {
  char *key_name = "Zhang Chu";
  char *key_name_ci = "zhang chu";

  // char **name_ptr = bsearch(&key_name, names,
  //                           sizeof names / sizeof *names,
  //                           sizeof *names,
  //                           (__compar_fn_t) strcmp); //
```

```
48 CompareStrsAddress
49
50   // char **name_ptr = bsearch(&key_name, names,
51   //                           sizeof names / sizeof *names,
52   //                           sizeof *names,
53   //                           CompareStrsAddress);
54
55   char **name_ptr = bsearch(&key_name, names,
56                             sizeof names / sizeof *names,
57                             sizeof *names,
58                             CompareStrs);
59
60   if (*name_ptr != NULL) {
61     printf("Found %s.\n", *name_ptr);
62   } else {
63     printf("Could not find %s.\n", key_name);
64   }
65
66   char **name_ci_ptr = bsearch(&key_name_ci, names,
67                                sizeof names / sizeof *names,
68                                sizeof *names,
69                                GetCompareFunction(false));
70   if (*name_ci_ptr != NULL) {
71     printf("Found %s.\n", *name_ci_ptr);
72   } else {
73     printf("Could not find %s.\n", key_name_ci);
74   }
75
76   return 0;
77 }
78
79 // Visualization: https://pythontutor.com/render.html#code=//%0A//%
   20Created%20by%20hfwei%20on%202023/12/13.%0A//%20Question%3A%20What%
   20if%20char%20key_name%5B%5D%20%3D%20%22Zhang%20Chu%22%3F%0A//%0A%0A%
   23include%20%3Cstdio.h%3E%0A%23include%20%3Cstring.h%3E%0A%0A//%20See
   %20https%3A//codebrowser.dev/glibc/glibc/stdlib/stdlib.h.html%
   23__compar_fn_t%0A//%20The%20first%20is%20a%20pointer%20to%20the%
   20key%20for%20the%20search,%0A//%20and%20the%20second%20is%20a%
   20pointer%20to%20the%20array%20element%20to%20be%20compared%20with%
   20the%20key.%0Atypedef%20int%20%28*__compar_fn_t%29%28const%20void%20
   *,%20const%20void%20*%29%3B%0A%0A//%20See%20https%3A//codebrowser.dev
   /glibc/glibc/bits/stdlib-bsearch.h.html%2319%0Avoid%20*bsearch%
   28const%20void%20*__key,%20const%20void%20*__base,%20size_t%20__nmemb
   ,%20size_t%20__size,%0A%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%
   20__compar_fn_t%20__compar%29%3B%0A%0Aint%20CompareStrs%28const%
   20void%20*left,%20const%20void%20*right%29%3B%0Aint%
   20CompareStrsAddress%28const%20char%20*left,%20const%20char%20*right%
   29%3B%0A%0Aconst%20char%20*names%5B%5D%20%3D%20%7B%0A%20%20%20%20%
   22Cui%20Jian%22,%0A%20%20%20%20%22Dou%20Wei%22,%0A%20%20%20%20%
   22ErShou%20Rose%22,%0A%20%20%20%20%22Hu%20Mage%22,%0A%20%20%20%20%
   22Li%20Zhi%22,%0A%20%20%20%20%22Luo%20Dayou%22,%0A%20%20%20%20%22Wan%
   20Qing%22,%0A%20%20%20%20%22Yao%22,%0A%20%20%20%20%22Zhang%20Chu%22,%
   0A%20%20%20%20%22ZuoXiao%22,%0A%7D%3B%0A%0Aint%20main%28void%29%20%7B
```

```
79 %0A%20%20char%20*key_name%20%3D%20%22Zhang%20Chu%22%3B%0A%0A%20%20//%
   20char%20**name_ptr%20%3D%20bsearch%28%26key_name,%20names,%0A%20%20
   //%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%
   20%20%20%20%20sizeof%20names%20/%20sizeof%20*names,%0A%20%20//%20%20%
   20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%
   20%20sizeof%20*names,%0A%20%20//%20%20%20%20%20%20%20%20%20%20%20%20%
   20%20%20%20%20%20%20%20%20%20%20%20%20%28__compar_fn_t%29%
   20strcmp%29%3B%20//%20CompareStrsAddress%0A%0A%20%20char%20**name_ptr
   %20%3D%20bsearch%28%26key_name,%20names,%0A%20%20%20%20%20%20%20%20%
   20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20sizeof%
   20names%20/%20sizeof%20*names,%0A%20%20%20%20%20%20%20%20%20%20%20%20%
   %20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20sizeof%20*names,%0A%
   20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%
   20%20%20%20%20CompareStrs%29%3B%0A%0A%20%20if%20%28*name_ptr%20!%3D%
   20NULL%29%20%7B%0A%20%20%20%20printf%28%22Found%20%25s.%5Cn%22,%20*
   name_ptr%29%3B%0A%20%20%7D%20else%20%7B%0A%20%20%20%20printf%28%
   22Could%20not%20find%20%25s.%5Cn%22,%20key_name%29%3B%0A%20%20%7D%0A%
   0A%20%20return%200%3B%0A%7D%0A%0Aint%20CompareStrs%28const%20void%20*
   left,%20const%20void%20*right%29%20%7B%0A%20%20char%20*const%20*pp1%
   20%3D%20left%3B%0A%20%20char%20*const%20*pp2%20%3D%20right%3B%0A%20%
   20return%20strcmp%28*pp1,%20*pp2%29%3B%0A%7D%0A%0A//%20What%20is%
   20the%20advantage%20of%20this%20version%3F%20%28performance%3F%3F%3F%
   29%0A//%20What%20is%20the%20disadvantage%20of%20this%20version%3F%20%
   28not%20flexible%3F%3F%3F%29%0Aint%20CompareStrsAddress%28const%
   20char%20*left,%20const%20char%20*right%29%20%7B%0A%20%20return%
   20strcmp%28left,%20right%29%3B%0A%7D%0A%0Avoid%20*bsearch%28const%
   20void%20*__key,%20const%20void%20*__base,%20size_t%20__nmemb,%
   20size_t%20__size,%0A%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%
   20__compar_fn_t%20__compar%29%20%7B%0A%20%20size_t%20__l,%20__u,%
   20__idx%3B%0A%20%20const%20void%20*__p%3B%0A%20%20int%20__comparison%
   3B%0A%20%20__l%20%3D%200%3B%0A%20%20__u%20%3D%20__nmemb%3B%0A%20%
   20while%20%28__l%20%3C%20__u%29%20%7B%0A%20%20%20%20__idx%20%3D%20%
   28__l%20%2B%20__u%29%20/%202%3B%0A%20%20%20%20__p%20%3D%20%28const%
   20void%20*%29%20%28%28%28const%20char%20*%29%20__base%29%20%2B%20%
   28__idx%20*%20__size%29%29%3B%0A%20%20%20%20__comparison%20%3D%20%28*
   __compar%29%28__key,%20__p%29%3B%0A%20%20%20%20if%20%28__comparison%
   20%3C%200%29%20%7B%0A%20%20%20%20%20%20__u%20%3D%20__idx%3B%0A%20%20%
   20%20%7D%20else%20if%20%28__comparison%20%3E%200%29%20%7B%0A%20%20%20
   %20%20%20__l%20%3D%20__idx%20%2B%201%3B%0A%20%20%20%20%7D%20else%20%
   7B%0A%20%20%20%20%20%20return%20%28void%20*%29%20__p%3B%0A%20%20%20%
   20%7D%0A%20%20%7D%0A%0A%20%20return%20NULL%3B%0A%7D&cppShowMemAddrs=
   true&cumulative=true&curInstr=14&heapPrimitives=nevernest&mode=
   display&origin=opt-frontend.js&py=c_gcc9.3.0&rawInputLstJSON=%5B%5D&
   textReferences=false
80 int CompareStrs(const void *left, const void *right) {
81   char *const *pp1 = left;
82   char *const *pp2 = right;
83   return strcmp(*pp1, *pp2);
84 }
85
86 int CompareStrsCI(const void *left, const void *right) {
87   const char *const *pp1 = left;
88   const char *const *pp2 = right;
```

```c
  // see https://www.ibm.com/docs/en/zos/2.4.0?topic=functions-strcasecmp-case-insensitive-string-comparison
  return strcasecmp(*pp1, *pp2);
}

// What is the advantage of this version? (performance???)
// What is the disadvantage of this version? (not flexible???)
// Visualization: https://pythontutor.com/render.html#code=//%0A//%20Created%20by%20hfwei%20on%202023/12/13.%0A//%20Question%3A%20What%20if%20char%20key_name%5B%5D%20%3D%20%22Zhang%20Chu%22%3F%0A//%0A%0A%23include%20%3Cstdio.h%3E%0A%23include%20%3Cstring.h%3E%0A%0A//%20See%20https%3A//codebrowser.dev/glibc/glibc/stdlib/stdlib.h.html%23__compar_fn_t%0A//%20The%20first%20is%20a%20pointer%20to%20the%20key%20for%20the%20search,%0A//%20and%20the%20second%20is%20a%20pointer%20to%20the%20array%20element%20to%20be%20compared%20with%20the%20key.%0Atypedef%20int%20%28*__compar_fn_t%29%28const%20void%20*,%20const%20void%20*%29%3B%0A%0A//%20See%20https%3A//codebrowser.dev/glibc/glibc/bits/stdlib-bsearch.h.html%2319%0Avoid%20*bsearch%28const%20void%20*__key,%20const%20void%20*__base,%20size_t%20__nmemb,%20size_t%20__size,%0A%20%20%20%20%20%20%20%20%20%20%20%20%20%20__compar_fn_t%20__compar%29%3B%0A%0Aint%20CompareStrs%28const%20void%20*left,%20const%20void%20*right%29%3B%0Aint%20CompareStrsAddress%28const%20char%20*left,%20const%20char%20*right%29%3B%0A%0Aconst%20char%20*names%5B%5D%20%3D%20%7B%0A%20%20%20%20%22Cui%20Jian%22,%0A%20%20%20%20%22Dou%20Wei%22,%0A%20%20%20%20%22ErShou%20Rose%22,%0A%20%20%20%20%22Hu%20Mage%22,%0A%20%20%20%20%22Li%20Zhi%22,%0A%20%20%20%20%22Luo%20Dayou%22,%0A%20%20%20%20%22Wan%20Qing%22,%0A%20%20%20%20%22Yao%22,%0A%20%20%20%20%22Zhang%20Chu%22,%0A%20%20%20%20%22ZuoXiao%22,%0A%7D%3B%0A%0Aint%20main%28void%29%20%7B%0A%20%20char%20*key_name%20%3D%20%22Zhang%20Chu%22%3B%0A%0A%20%20//%20char%20**name_ptr%20%3D%20bsearch%28%26key_name,%20names,%0A%20%20//%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20sizeof%20names%20/%20sizeof%20*names,%0A%20%20//%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20sizeof%20*names,%0A%20%20//%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%28__compar_fn_t%29%20strcmp%29%3B%20//%20CompareStrsAddress%0A%0A%20%20char%20**name_ptr%20%3D%20bsearch%28%26key_name,%20names,%0A%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20sizeof%20names%20/%20sizeof%20*names,%0A%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20sizeof%20*names,%0A%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20CompareStrsAddress%29%3B%0A%0A%20%20if%20%28*name_ptr%20!%3D%20NULL%29%20%7B%0A%20%20%20%20printf%28%22Found%20%25s.%5Cn%22,%20*name_ptr%29%3B%0A%20%20%7D%20else%20%7B%0A%20%20%20%20printf%28%22Could%20not%20find%20%25s.%5Cn%22,%20key_name%29%3B%0A%20%20%7D%0A%0A%20%20return%200%3B%0A%7D%0A%0Aint%20CompareStrs%28const%20void%20*left,%20const%20void%20*right%29%20%7B%0A%20%20char%20*const%20*pp1%20%3D%20left%3B%0A%20%20char%20*const%20*pp2%20%3D%20right%3B%0A%20%20return%20strcmp%28*pp1,%20*pp2%29%3B%0A%7D%0A%0A//%20What%20is%20the%20advantage%20of%20this%20version%3F%20%28performance%3F%3F%3F%29%0A//%20What%20is%20the%20disadvantage%20of%20this%20version%3F%20%28not%20flexible%3F%3F%3F%29%0Aint%20CompareStrsAddress%28const%
```

```
95 20char%20*left,%20const%20char%20*right%29%20%7B%0A%20%20return%
   20strcmp%28left,%20right%29%3B%0A%7D%0A%0Avoid%20*bsearch%28const%
   20void%20*__key,%20const%20void%20*__base,%20size_t%20__nmemb,%
   20size_t%20__size,%0A%20%20%20%20%20%20%20%20%20%20%20%20%20%
   20__compar_fn_t%20__compar%29%20%7B%0A%20%20size_t%20__l,%20__u,%
   20__idx%3B%0A%20%20const%20void%20*__p%3B%0A%20%20int%20__comparison%
   3B%0A%20%20__l%20%3D%200%3B%0A%20%20__u%20%3D%20__nmemb%3B%0A%20%
   20while%20%28__l%20%3C%20__u%29%20%7B%0A%20%20%20%20__idx%20%3D%20%
   28__l%20%2B%20__u%29%20/%202%3B%0A%20%20%20%20__p%20%3D%20%28const%
   20void%20*%29%20%28%28const%20char%20*%29%20__base%29%20%2B%20%
   28__idx%20*%20__size%29%29%3B%0A%20%20%20%20__comparison%20%3D%20%28*
   __compar%29%28__key,%20__p%29%3B%0A%20%20%20%20if%20%28__comparison%
   20%3C%200%29%20%7B%0A%20%20%20%20%20%20__u%20%3D%20__idx%3B%0A%20%20%
   20%20%7D%20else%20if%20%28__comparison%20%3E%200%29%20%7B%0A%20%20%20
   %20%20%20__l%20%3D%20__idx%20%2B%201%3B%0A%20%20%20%20%7D%20else%20%
   7B%0A%20%20%20%20%20%20return%20%28void%20*%29%20__p%3B%0A%20%20%20%
   20%7D%0A%20%20%7D%0A%0A%20%20return%20NULL%3B%0A%7D&cppShowMemAddrs=
   true&cumulative=true&curInstr=30&heapPrimitives=nevernest&mode=
   display&origin=opt-frontend.js&py=c_gcc9.3.0&rawInputLstJSON=%5B%5D&
   textReferences=false
96 int CompareStrsAddress(const char *left, const char *right) {
97   return strcmp(left, right);
98 }
99
100 void *bsearch(const void *__key, const void *__base, size_t __nmemb,
    size_t __size,
101              __compar_fn_t __compar) {
102   size_t __l, __u, __idx;
103   const void *__p;
104   int __comparison;
105   __l = 0;
106   __u = __nmemb;
107   while (__l < __u) {
108     __idx = (__l + __u) / 2;
109     __p = (const void *) (((const char *) __base) + (__idx * __size
    ));
110     __comparison = (*__compar)(__key, __p);
111     if (__comparison < 0) {
112       __u = __idx;
113     } else if (__comparison > 0) {
114       __l = __idx + 1;
115     } else {
116       return (void *) __p;
117     }
118   }
119
120   return NULL;
121 }
122
123 void *bsearch_leftmost(const void *__key, const void *__base, size_t
    __nmemb, size_t __size,
124                        __compar_fn_t __compar) {
125   size_t __l, __u, __idx;
```

```
126   const void *__p;
127   int __comparison;
128   __l = 0;
129   __u = __nmemb;
130   while (__l < __u) {
131     __idx = (__l + __u) / 2;
132     __p = (const void *) (((const char *) __base) + (__idx * __size
    ));
133     __comparison = (*__compar)(__key, __p);
134     if (__comparison < 0) {
135       __u = __idx;
136     } else if (__comparison > 0) {
137       __l = __idx + 1;
138     } else {
139       return (void *) __p;
140     }
141   }
142
143   return NULL;
144 }
```