```c
// Created by hengxin on 2024/12/04.

int main() {
  char **argv;

  int *names[10];

  int (*musician_score_table)[10];

  int *StrCpyStd(char *dest, const char *src);

  int (*comp)(const void *left, const void *right);

  // see https://en.cppreference.com/w/c/program/atexit
  int atexit(void (*func)(void));

  // see https://en.cppreference.com/w/c/program/signal
  void (*signal(int sig, void (*handler)(int)))(int);

  char (*(*func(int num, char *str))[3])(void);

  char (*(*arr[3])(void))[5];

  // Refer to https://cdecl.org/ for more practice.
  // See https://c-faq.com/decl/spiral.anderson.html for secrets!!!
}
```

```c
// Created by hengxin on 2024/12/04.

#include <stdio.h>
#include <stdlib.h>
#include <limits.h>
#include <string.h>

// void qsort( void *ptr, size_t count, size_t size,
//             int (*comp)(const void *, const void *) );

typedef int (*CompareFunction)(const void *, const void *);

int CompareInts(const void *left, const void *right);
int CompareStrs(const void *left, const void *right);
int CompareStrsWrong(const void *left, const void *right);

void PrintInts(const int *integers, size_t len);
void PrintStrs(const char *str[], size_t len);

int main() {
  int integers[] = {-2, 99, 0, -743, 2, INT_MIN, 4};
  int size_of_integers = sizeof integers / sizeof *integers;

  CompareFunction comp = CompareInts;
  // int (*comp)(const void *, const void *) = CompareInts;

  qsort(integers, size_of_integers, sizeof integers[0],
        comp);
  PrintInts(integers, size_of_integers);

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

```c
  };
  size_t size_of_names = sizeof names / sizeof *names;

  comp = CompareStrs;
  qsort(names, size_of_names, sizeof names[0],
        comp);
  PrintStrs(names, size_of_names);
}

// left: int *
int CompareInts(const void *left, const void *right) {
  int left_int = *(const int *) left;
  int right_int = *(const int *) right;

  return (left_int > right_int) - (left_int < right_int);
}

// left, right: char **
int CompareStrs(const void *left, const void *right) {
  const char *const *pp1 = left;
  const char *const *pp2 = right;

  return strcmp(*pp1, *pp2);
}

int CompareStrsWrong(const void *left, const void *right
) {
  return strcmp(left, right);
}

void PrintInts(const int *integers, size_t len) {
  printf("\n");
  for (int i = 0; i < len; i++) {
    printf("%d ", integers[i]);
  }
  printf("\n");
}

void PrintStrs(const char *str[], size_t len) {
  printf("\n");
  for (int i = 0; i < len; i++) {
    printf("%s\n", str[i]);
  }
  printf("\n");
```

```
85 }
```

```c
// Created by hfwei on 2024/12/04.
// See https://en.cppreference.com/w/c/program/atexit

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

void f1(void) {
  puts("f1");
}

void f2(void) {
  puts("f2");
}

int main(void) {
  if (!atexit(f1) && !atexit(f2) && !atexit(f2)) {
    return EXIT_SUCCESS;
  }

  // atexit registration failed
  return EXIT_FAILURE;

}   // <- if registration was successful calls f2, f2, f1
```

```c
// Created by hfwei on 2024/12/04.

#include <stdio.h>
#include <signal.h>

void SIGSEGV_Handler(int sig) {
  printf("SIGSEGV %d is caught.\n", sig);
}

int main(void) {
  signal(SIGSEGV, SIGSEGV_Handler);
  raise(SIGSEGV);

  // int *p = NULL;
  // *p = 0;

  return 0;
}
```

```c
// Created by hfwei on 2024/12/04.

#include <stdio.h>

// See https://elixir.bootlin.com/linux/latest/source/include/linux/types.h#L245
typedef int (*cmp_func_t)(const void *a, const void *b);

// See https://elixir.bootlin.com/linux/latest/source/include/linux/bsearch.h#L8
void *bsearch(const void *key, const void *base,
              size_t num, size_t size, cmp_func_t cmp);

int main(void) {

  return 0;
}

void *bsearch(const void *key, const void *base, size_t num, size_t size, cmp_func_t cmp) {
  const char *pivot;
  int result;

  while (num > 0) {
    pivot = base + (num >> 1) * size;
    result = cmp(key, pivot);

    if (result == 0) {
      return (void *) pivot;
    }

    if (result > 0) {
      base = pivot + size;
      num--;
    }

    num >>= 1;
  }

  return NULL;
}
```

```
1 # `11-function-pointers`
2
3 ## `integrate.c`
4
5 ## `sort.c`
6
7 ## `bsearch-gnuc.c`
8
9 ## `decl.c`
```

```c
// Created by hfwei on 2024/12/04.
// A nice function pointer example on Riemann integration:
// https://en.wikipedia.org/wiki/Function_pointer

#include <stdio.h>
#include <math.h>

#define NUM_OF_PARTITIONS 100000

double Integrate(double low, double high, double (*func)(double));
double Square(double x);

int main() {
  double low = 0.0;
  double high = 1.0;
  double integration = 0.0;

  integration = Integrate(low, high, sin);
  printf("Sin(0,0 .. 1.0) = %f\n", integration);

  integration = Integrate(low, high, cos);
  printf("Cos(0,0 .. 1.0) = %f\n", integration);

  integration = Integrate(low, high, Square);
  printf("Square(0,0 .. 1.0) = %f\n", integration);

  double (*funcs[])(double) = {sin, cos, Square};
  for (int i = 0; i < 3; ++i) {
    printf("integration: %f\n",
           Integrate(low, high, funcs[i]));
  }

  return 0;
}

double Integrate(double low, double high, double (*func)(double)) {
  double interval = (high - low) / (double)NUM_OF_PARTITIONS;

  double sum = 0.0;
  for (int i = 0; i < NUM_OF_PARTITIONS; ++i) {
    double xi = low + interval * i;
```

```c
    double yi = func(xi);
    sum += yi * interval;
  }

  return sum;
}

double Square(double x) {
  return x * x;
}
```

```c
// Created by hfwei on 2024/12/04.
// 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
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 CompareStrsAddress(const void *left, const void *right);
int CompareStrsCI(const void *left, const void *right);

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

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

  char **name_ptr = bsearch(&key_name,
                            names, sizeof names / sizeof names[0],
```

```
41                                sizeof names[0],
42                                CompareStrs);
43    if (name_ptr != NULL) {
44      printf("Found\n");
45    } else {
46      printf("Not Found\n");
47    }
48
49    return 0;
50  }
51
52  // 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%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%
    20sizeof%20*names,%0A%20%20//%20%20%20%20%20%20%20%20%20%
```

```
52 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%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%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__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&
```

```
52 rawInputLstJSON=%5B%5D&textReferences=false
53 int CompareStrs(const void *left, const void *right) {
54   char *const *pp1 = left;
55   char *const *pp2 = right;
56   return strcmp(*pp1, *pp2);
57 }
58
59 // case insensitive
60 int CompareStrsCI(const void *left, const void *right) {
61   char *const *pp1 = left;
62   char *const *pp2 = right;
63   return strcasecmp(*pp1, *pp2);
64 }
65
66 // 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%20%20sizeof%20names%20/%
```

```
66 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%
   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%
   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%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%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%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%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%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%
```

```
 66 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
 67 int CompareStrsAddress(const void *left, const void *
    right) {
 68   return strcmp(left, right);
 69 }
 70
 71 void *bsearch(const void *__key, const void *__base,
    size_t __nmemb, size_t __size,
 72               __compar_fn_t __compar) {
 73   size_t __l, __u, __idx;
 74   const void *__p;
 75   int __comparison;
 76   __l = 0;
 77   __u = __nmemb;
 78   while (__l < __u) {
 79     __idx = (__l + __u) / 2;
 80     __p = (const void *) (((const char *) __base) + (
    __idx * __size));
 81     __comparison = (*__compar)(__key, __p);
 82     if (__comparison < 0) {
 83       __u = __idx;
 84     } else if (__comparison > 0) {
 85       __l = __idx + 1;
 86     } else {
 87       return (void *) __p;
 88     }
 89   }
 90
 91   return NULL;
 92 }
 93
 94 // void *bsearch_leftmost(const void *__key, const void *
    __base,
 95 //                        size_t __nmemb, size_t __size,
 96 //                        __compar_fn_t __compar) {
 97 //   size_t __l, __u, __idx;
 98 //   const void *__p;
 99 //   int __comparison;
100 //
101 //   __l = 0;
```

```
102 //   __u = __nmemb;
103 //   // added by ant
104 //   void *__index = NULL;
105 //
106 //   while (__l < __u) {
107 //     __idx = (__l + __u) / 2;
108 //     __p = (const void *) (((const char *) __base) + (
    __idx * __size));
109 //     __comparison = (*__compar)(__key, __p);
110 //     if (__comparison < 0) {
111 //       __u = __idx;
112 //     } else if (__comparison > 0) {
113 //       __l = __idx + 1;
114 //     } else {
115 //       // added by ant
116 //       __index = (void *) __p;
117 //       __u = __idx - 1;
118 //     }
119 //   }
120 //
121 //   // added by ant
122 //   return __index;
123 // }
```

```
add_executable(integrate integrate.c)

add_executable(sort sort.c)

add_executable(bsearch bsearch.c)
add_executable(bsearch-gnuc bsearch-gnuc.c)

add_executable(11-decl decl.c)
add_executable(atexit atexit.c)
add_executable(signal signal.c)
```