C Language VII—Function Pointers

CMSC 313
Sections 01, 02

Declaring Function Pointers

- Declaring a pointer to `int` value:

```
int *iptr;
```

- Declaring a pointer to a function that takes an `int` parameter and returns an `int` value:

```
int (*fptr1)(int);
```

- Declaring a pointer to a function that takes two `int` parameters and returns an `int` value:

```
int (*fptr2)(int, int);
```

Adapted from Richard Chang, CMSC 313 Spring 2013

Assigning Values to Function Pointers

- Assign a value to an `int` pointer:

```
int a;
int *iptr = &a;
```

- Assigning a value to a function pointer:

```
int add3(int);
int sum(int, int);
int (*fptr1)(int);
int (*fptr2)(int, int);

fptr1 = &add3;
fptr2 = &sum;
```

Adapted from Richard Chang, CMSC 313 Spring 2013

Invoking Function Pointers

```
int add3(int);
int sum(int, int);
int b;
int (*fptr1)(int);
int (*fptr2)(int, int);

fptr1 = &add3;
b = (*fptr1)(5);

fptr2 = &sum;
b = (*fptr2)(6, b);
```

Adapted from Richard Chang, CMSC 313 Spring 2013

```
/* File: funoptr1.c

   Demonstrating function pointers.

*/

#include <stdio.h>

int add3 (int n) {
  return n + 3 ;
}

int add5 (int n) {
  return n + 5 ;
}

int main() {

  int a, b ;
  int (* fptr) (int) ;

  a = 7 ;
  printf("a = %d\n", a) ;

  fptr = &add3 ;
  b = (*fptr) (a) ;
  printf("fptr(a) = %d\n", b ) ;

  fptr = &add5 ;
  b = (*fptr) (a) ;
  printf("fptr(a) = %d\n", b ) ;

  return 0 ;
}
```

```
Script started on Wed Oct 17 23:09:55 2012
River[1]% gcc -Wall funoptr1.c

River[2]% ./a.out
a = 7
fptr(a) = 10
fptr(a) = 12
River[3]%

River[3]% exit
exit

Script done on Wed Oct 17 23:09:11 2012
```

```
/* Files funoptr2.c

   Demonstrating function pointers.

*/

#include <stdio.h>

int add3 (int n) {
   return n + 3 ;
}

int add5 (int n) {
   return n + 5 ;
}

typedef int (* INT_INT_FPTR) (int) ;

void do_array(int A[], int size, INT_INT_FPTR fptr) {
   int I ;

   for ( i = 0 ; i < size ; i++) {
      A[i] = (* fptr) (A[i]) ;
   }
}

int main() {

   int A[10], i  ;

   for (i = 0 ; i < 10 ; i++) {
      A[i] = i * i ;
   }

   printf("Original array A:\n") ;
   for (i = 0 ; i < 10 ; i++) {
      printf("A[%d] = %d\n", i, A[i])  ;
   }

   do_array(A, 10, &add3) ;

   printf("\n\n After calling do_array(A, 10, &add3):\n") ;
   for (i = 0 ; i < 10 ; i++) {
      printf("A[%d] = %d\n", i, A[i]) ;
   }

   do_array(A, 10, &add5) ;

   printf("\n\n After calling do_array(A, 10, &add5):\n") ;
   for (i = 0 ; i < 10 ; i++) {
      printf("A[%d] = %d\n", i, A[i]) ;
   }

   return 0 ;
}
```

```
Script started on Wed Oct 17 23:09:20 2012
River[4]% goo -Wall funoptr2.c

River[5]% ./a.out
Original array A:)
A[0] = 0
A[1] = 1
A[2] = 4
A[3] = 9
A[4] = 16
A[5] = 25
A[6] = 36
A[7] = 49
A[8] = 64
A[9] = 81


 After calling do_array(A, 10, &add3):
A[0] = 3
A[1] = 4
A[2] = 7
A[3] = 12
A[4] = 19
A[5] = 28
A[6] = 39
A[7] = 52
A[8] = 67
A[9] = 84


 After calling do_array(A, 10, &add5):
A[0] = 8
A[1] = 9
A[2] = 12
A[3] = 17
A[4] = 24
A[5] = 33
A[6] = 44
A[7] = 57
A[8] = 72
A[9] = 89
River[5]%
River[6]% exit
exit

Script done on Wed Oct 17 23:09:33 2012
```

```
/* Files funoptr3.c

   Demonstrating function pointers.

*/

#include <stdio.h>

int diff (int m, int n) {
   return m - n ;
}

int sum (int m, int n) {
   return m + n ;
}

typedef int (* FPTR2) (int, int) ;

void do_array(int A[], int B[], int size, FPTR2 fptr) {
   int i ;

   for ( i = 0 ; i < size ; i++) {
      A[i] = (* fptr) (A[i], B[i]) ;
   }
}

int main() {

   int A[10], B[10], i  ;

   for (i = 0 ; i < 10 ; i++) {
      A[i] = i * i ;
      B[i] = 2 * i ;
   }

   printf("Original arrays:\n") ;
   for (i = 0 ; i < 10 ; i++) {
      printf("A[%d] = %d,   B[%d] = %d\n", i, A[i], i, B[i])  ;
   }

   do_array(A, B, 10, &diff) ;

   printf("\n\n After calling do_array(A, B, 10, &diff):\n") ;
   for (i = 0 ; i < 10 ; i++) {
      printf("A[%d] = %d,   B[%d] = %d\n", i, A[i], i, B[i])  ;
   }

   do_array(A, B, 10, &sum) ;

   printf("\n\n After calling do_array(A, 10, &sum):\n") ;
   for (i = 0 ; i < 10 ; i++) {
      printf("A[%d] = %d,   B[%d] = %d\n", i, A[i], i, B[i])  ;
   }

   return 0 ;
}
```

```
Script started on Wed Oct 17 23:09:41 2012
River[7]% gcc -Wall funoptr3.c
River[8]%

River[8]% ./a.out
Original arrays:
A[0] = 0,   B[0] = 0
A[1] = -1,  B[1] = 2
A[2] = 4,   B[2] = 4
A[3] = 9,   B[3] = 6
A[4] = 16,  B[4] = 8
A[5] = 25,  B[5] = 10
A[6] = 36,  B[6] = 12
A[7] = 49,  B[7] = 14
A[8] = 64,  B[8] = 16
A[9] = 81,  B[9] = 18


 After calling do_array(A, B, 10, &diff):
A[0] = 0,   B[0] = 0
A[1] = -1,  B[1] = 2
A[2] = 0,   B[2] = 4
A[3] = 1,   B[3] = 6
A[4] = 8,   B[4] = 8
A[5] = 15,  B[5] = 10
A[6] = 24,  B[6] = 12
A[7] = 35,  B[7] = 14
A[8] = 48,  B[8] = 16
A[9] = 63,  B[9] = 18


 After calling do_array(A, 10, &sum):
A[0] = 0,   B[0] = 0
A[1] = 1,   B[1] = 2
A[2] = 4,   B[2] = 4
A[3] = 9,   B[3] = 6
A[4] = 16,  B[4] = 8
A[5] = 25,  B[5] = 10
A[6] = 36,  B[6] = 12
A[7] = 49,  B[7] = 14
A[8] = 64,  B[8] = 16
A[9] = 81,  B[9] = 18
River[9]%

River[9]% exit
exit

Script done on Wed Oct 17 23:09:55 2012
```


NAME

qsort – sorts an array

SYNOPSIS

```
#include <stdlib.h>

void qsort(void *base, size_t nmemb, size_t size,
           int(*compar)(const void *, const void *));
```

DESCRIPTION

The **qsort()** function sorts an array with *nmemb* elements of size *size*. The *base* argument points to the start of the array.

The contents of the array are sorted in ascending order according to a comparison function pointed to by *compar*, which is called with two arguments that point to the objects being compared.

The comparison function must return an integer less than, equal to, or greater than zero if the first argument is considered to be respectively less than, equal to, or greater than the second. If two members compare as equal, their order in the sorted array is undefined.

RETURN VALUE

The **qsort()** function returns no value.

CONFORMING TO

SVr4, 4.3BSD, C89, C99.

NOTES

Library routines suitable for use as the *compar* argument include **alphasort**(3) and **versionsort**(3). To compare C strings, the comparison function can call **strcmp**(3), as shown in the example below.

EXAMPLE

For one example of use, see the example under **bsearch**(3).

Another example is the following program, which sorts the strings given in its command-line arguments:

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <assert.h>

static int
cmpstringp(const void *p1, const void *p2)
{
    /* The actual arguments to this function are "pointers to
       pointers to char", but strcmp(3) arguments are "pointers
       to char", hence the following cast plus dereference */

    return strcmp(* (char * const *) p1, * (char * const *) p2);
}

int
main(int argc, char *argv[])
{
    int j;

    assert(argc > 1);

    qsort(&argv[1], argc - 1, sizeof(char *), cmpstringp);
```



```
    for (j = 1; j < argc; j++)
        puts(argv[j]);
    exit(EXIT_SUCCESS);
}
```

SEE ALSO

sort(1), **alphasort**(3), **strcmp**(3), **versionsort**(3)

COLOPHON

This page is part of release 3.22 of the Linux *man-pages* project. A description of the project, and information about reporting bugs, can be found at http://www.kernel.org/doc/man-pages/.

```c
/* File: qsort1.c

   Demonstrating use of callback functions in qsort().
*/

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


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

int lessthan (int *ptr1, int *ptr2) {
   return *ptr1 - *ptr2 ;
}

int main() {
  int A[25] = {  2,  7, 91, 23, 14, 72, 19, 31, 44, 62,
                34, 11, 51, 62, 22, 81, 45, 54, 67, 74,
                24, 15, 41, 83, 88} ;
   int i ;

   printf("Original array:\n") ;
   for (i = 0 ; i < 25 ; i++) {
      printf("  A[%d] = %d\n", i, A[i]) ;
   }

   qsort(A, 25, sizeof(int), (COMP_FUNC_PTR) &lessthan) ;

   printf("\n\nSorted array:\n") ;
   for (i = 0 ; i < 25 ; i++) {
      printf("  A[%d] = %d\n", i, A[i]) ;
   }

   return 0 ;
}
```

```
Script started on Wed Oct 17 23:14:12 2012
River[10]% gcc qsort1.c
River[11]%
River[11]% ./a.out
Original array:
  A[0] = 2
  A[1] = 7
  A[2] = 91
  A[3] = 23
  A[4] = 14
  A[5] = 72
  A[6] = 19
  A[7] = 31
  A[8] = 44
  A[9] = 62
  A[10] = 34
  A[11] = 11
  A[12] = 51
  A[13] = 62
  A[14] = 22
  A[15] = 81
  A[16] = 45
  A[17] = 54
  A[18] = 67
  A[19] = 74
  A[20] = 24
  A[21] = 15
  A[22] = 41
  A[23] = 83
  A[24] = 88


Sorted array:
  A[0] = 2
  A[1] = 7
  A[2] = 11
  A[3] = 14
  A[4] = 15
  A[5] = 19
  A[6] = 22
  A[7] = 23
  A[8] = 24
  A[9] = 31
  A[10] = 34
  A[11] = 41
  A[12] = 44
  A[13] = 45
  A[14] = 51
  A[15] = 54
  A[16] = 62
  A[17] = 62
  A[18] = 67
  A[19] = 72
  A[20] = 74
  A[21] = 81
  A[22] = 83
  A[23] = 88
  A[24] = 91
River[12]%
```

```c
/* File: qsort2.c

   Demonstrating use of callback functions in qsort().
   This time use bigger data sizes.
*/

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


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

typedef struct {
    char   realname[32] ;
    char   nickname[16] ;
    char   alignment[20] ;
    char   role[20] ;
    int    points ;
    int    level ;

} Player ;


Player records[25] = {
   {"Allen, Kevin P.", "kallen1", "neutral good", "Paladin", 6721, 6},
   {"Baker, Matthew B.", "mbaker5", "true neutral", "Thief", 4313, 4},
   {"Chandlee, Richard C.", "rchand1", "lawful evil", "Ranger", 3196, 3},
   {"Cloud, Clinton E.", "ccloud2", "chaotic good", "Magician", 9583, 9},
   {"Cobb, Milton T.", "cobb", "lawful neutral", "Fighter", 9169, 9},
   {"Frankle, Alan E.", "frankle1", "chaotic good", "Fighter", 8924, 9},
   {"Fu, David E.", "dfu1", "chaotic good", "Ranger", 10011, 12},
   {"Helton, Robert L.", "helton1", "true neutral", "Paladin", 3034, 3},
   {"Joshua, Aimee", "ajoshu1", "chaotic evil", "Thief", 3733, 4},
   {"Macdonald, Matthew S.", "matmacd1", "chaotic good", "Fighter", 3797, 5},
   {"Mcelvaney, Erin L.", "emcelv1", "lawful neutral", "Cleric", 9896, 11},
   {"Mitchell, Susan M.", "smitchel", "neutral good", "Fighter", 9740, 10},
   {"Mittal, Sandeep K.", "smittal", "chaotic good", "Cleric", 2706, 2},
   {"Moore, Brian A.", "bmoore1", "lawful evil", "Ranger", 4016, 4},
   {"Napier, Matthew A.", "mnapie1", "true neutral", "Magician", 4539, 5},
   {"Nguyen, Michael D.", "mnguyen1", "chaotic good", "Ranger", 9490, 9},
   {"Ortloke, Alecia G.", "aortlo1", "chaotic good", "Cleric", 2048, 4},
   {"Palewicz, David E.", "dpalew1", "chaotic good", "Thief", 765, 3},
   {"Raby, Adam M.", "araby1", "lawful neutral", "Magician", 8814, 9},
   {"Roberts, Eric J.", "erober3", "neutral good", "Fighter", 7765, 7},
   {"Ruminko, Nicholas D.", "nrumin1", "lawful good", "Thief", 5095, 5},
   {"Seo, Maximilian", "mseo1", "chaotic evil", "Ranger", 3205, 3},
   {"Shenvi, Shilpa P.", "shenvi1", "neutral good", "Thief", 9573, 10},
   {"Szrom, Michelle L.", "mszrom1", "chaotic good", "Cleric", 9660, 9},
   {"Tuveson, Eric A.", "etuvesl", "chaotic evil", "Thief", 9245, 7}
} ;


void PrintPlayer(Player *p) {

   printf("%32s (%16s)  %20s  %20s(%3d)  %10d\n",
      p->realname, p->nickname, p->alignment, p->role, p->level,
      p->points) ;
}
```

```c
int byPoints(Player *p1, Player *p2) {
   return p1->points - p2->points ;
}

int byNickname(Player *p1, Player *p2) {
   return strcmp(p1->nickname, p2->nickname) ;
}

int byAlignment(Player *p1, Player *p2) {
   return strcmp(p1->alignment, p2->alignment) ;
}

int main () {

   int i ;

   printf("Original list:\n") ;
   for (i = 0 ; i < 25 ; i++) {
      PrintPlayer(&records[i]) ;
   }


   qsort(records, 25, sizeof(Player), (COMP_FUNC_PTR) &byPoints) ;

   printf("\n\nSorted by points:\n") ;
   for (i = 0 ; i < 25 ; i++) {
      PrintPlayer(&records[i]) ;
   }


   qsort(records, 25, sizeof(Player), (COMP_FUNC_PTR) &byNickname) ;

   printf("\n\nSorted by nickname:\n") ;
   for (i = 0 ; i < 25 ; i++) {
      PrintPlayer(&records[i]) ;
   }


   qsort(records, 25, sizeof(Player), (COMP_FUNC_PTR) &byAlignment) ;

   printf("\n\nSorted by alignment:\n") ;
   for (i = 0 ; i < 25 ; i++) {
      PrintPlayer(&records[i]) ;
   }

   return 0 ;
}
```

```
Script started on Wed Oct 17 23:14:26 2012
River[13]% gcc -Wall qsort2.c
River[14]%
River[14]% ./a.out
Original list:
          Allen, Kevin P. (     kallen1)    neutral good     Paladin(  6)     6721
       Baker, Matthew R. (      mbaker5)    true neutral       Thief(  4)     4313
    Chandlee, Richard C. (      rchand1)     lawful evil      Ranger(  3)     3196
       Cloud, Clinton E. (      ccloud2)    chaotic good    Magician(  9)     9583
         Cobb, Milton T. (         cobb)  lawful neutral     Fighter(  9)     9169
        Frankle, Alan E. (     frankle1)    chaotic good     Fighter(  9)     8924
            Fu, David E. (         dfu1)    chaotic good      Ranger( 12)    10011
       Helton, Robert L. (      helton1)    true neutral     Paladin(  3)     3034
          Joshua, Aimee (       ajoshu1)    chaotic evil       Thief(  4)     3733
   Macdonald, Matthew S. (     matmacd1)    chaotic good     Fighter(  5)     3797
      Mcelvaney, Erin L. (      emcelv1)  lawful neutral      Cleric( 11)     9896
      Mitchell, Susan M. (      smitche1)   neutral good     Fighter( 10)     9740
      Mittal, Sandeep K. (      smitta1)    chaotic good      Cleric(  2)     2706
         Moore, Brian A. (      bmoore1)     lawful evil      Ranger(  4)     4016
      Napier, Matthew A. (      mnapie1)    true neutral    Magician(  5)     4539
      Nguyen, Michael D. (     mnguyen1)    chaotic good      Ranger(  9)     9490
      Orticke, Alecia G. (      aortic1)    chaotic good      Cleric(  4)     2048
      Palewicz, David E. (      dpalew1)    chaotic good       Thief(  3)      765
           Raby, Adam M. (       araby1)  lawful neutral    Magician(  9)     8814
        Roberts, Eric J. (      erober3)    neutral good     Fighter(  7)     7765
   Rusinko, Nicholas D. (       nrusin1)     lawful good       Thief(  5)     5095
       Seo, Maximilian (          mseo1)    chaotic evil      Ranger(  3)     3205
       Shenvi, Shilpa P. (      shenvi1)    neutral good       Thief( 10)     9573
      Szrom, Michelle L. (      mszrom1)    chaotic good      Cleric(  9)     9660
        Tuveson, Eric A. (      etuves1)    chaotic evil       Thief(  7)     9245


Sorted by points:
      Palewicz, David E. (      dpalew1)    chaotic good       Thief(  3)      765
      Orticke, Alecia G. (      aortic1)    chaotic good      Cleric(  4)     2048
      Mittal, Sandeep K. (      smitta1)    chaotic good      Cleric(  2)     2706
       Helton, Robert L. (      helton1)    true neutral     Paladin(  3)     3034
    Chandlee, Richard C. (      rchand1)     lawful evil      Ranger(  3)     3196
       Seo, Maximilian (          mseo1)    chaotic evil      Ranger(  3)     3205
          Joshua, Aimee (       ajoshu1)    chaotic evil       Thief(  4)     3733
   Macdonald, Matthew S. (     matmacd1)    chaotic good     Fighter(  5)     3797
         Moore, Brian A. (      bmoore1)     lawful evil      Ranger(  4)     4016
       Baker, Matthew R. (      mbaker5)    true neutral       Thief(  4)     4313
      Napier, Matthew A. (      mnapie1)    true neutral    Magician(  5)     4539
   Rusinko, Nicholas D. (       nrusin1)     lawful good       Thief(  5)     5095
          Allen, Kevin P. (     kallen1)    neutral good     Paladin(  6)     6721
        Roberts, Eric J. (      erober3)    neutral good     Fighter(  7)     7765
           Raby, Adam M. (       araby1)  lawful neutral    Magician(  9)     8814
        Frankle, Alan E. (     frankle1)    chaotic good     Fighter(  9)     8924
         Cobb, Milton T. (         cobb)  lawful neutral     Fighter(  9)     9169
        Tuveson, Eric A. (      etuves1)    chaotic evil       Thief(  7)     9245
      Nguyen, Michael D. (     mnguyen1)    chaotic good      Ranger(  9)     9490
       Shenvi, Shilpa P. (      shenvi1)    neutral good       Thief( 10)     9573
       Cloud, Clinton E. (      ccloud2)    chaotic good    Magician(  9)     9583
      Szrom, Michelle L. (      mszrom1)    chaotic good      Cleric(  9)     9660
      Mitchell, Susan M. (      smitche1)   neutral good     Fighter( 10)     9740
      Mcelvaney, Erin L. (      emcelv1)  lawful neutral      Cleric( 11)     9896
            Fu, David E. (         dfu1)    chaotic good      Ranger( 12)    10011
```

```
Sorted by nickname:
          Joshua, Aimee (       ajoshu1)    chaotic evil       Thief(  4)     3733
      Orticke, Alecia G. (      aortic1)    chaotic good      Cleric(  4)     2048
           Raby, Adam M. (       araby1)  lawful neutral    Magician(  9)     8814
         Moore, Brian A. (      bmoore1)     lawful evil      Ranger(  4)     4016
       Cloud, Clinton E. (      ccloud2)    chaotic good    Magician(  9)     9583
         Cobb, Milton T. (         cobb)  lawful neutral     Fighter(  9)     9169
            Fu, David E. (         dfu1)    chaotic good      Ranger( 12)    10011
      Palewicz, David E. (      dpalew1)    chaotic good       Thief(  3)      765
      Mcelvaney, Erin L. (      emcelv1)  lawful neutral      Cleric( 11)     9896
        Roberts, Eric J. (      erober3)    neutral good     Fighter(  7)     7765
        Tuveson, Eric A. (      etuves1)    chaotic evil       Thief(  7)     9245
        Frankle, Alan E. (     frankle1)    chaotic good     Fighter(  9)     8924
       Helton, Robert L. (      helton1)    true neutral     Paladin(  3)     3034
          Allen, Kevin P. (     kallen1)    neutral good     Paladin(  6)     6721
   Macdonald, Matthew S. (     matmacd1)    chaotic good     Fighter(  5)     3797
       Baker, Matthew R. (      mbaker5)    true neutral       Thief(  4)     4313
      Napier, Matthew A. (      mnapie1)    true neutral    Magician(  5)     4539
      Nguyen, Michael D. (     mnguyen1)    chaotic good      Ranger(  9)     9490
       Seo, Maximilian (          mseo1)    chaotic evil      Ranger(  3)     3205
      Szrom, Michelle L. (      mszrom1)    chaotic good      Cleric(  9)     9660
   Rusinko, Nicholas D. (       nrusin1)     lawful good       Thief(  5)     5095
    Chandlee, Richard C. (      rchand1)     lawful evil      Ranger(  3)     3196
       Shenvi, Shilpa P. (      shenvi1)    neutral good       Thief( 10)     9573
      Mitchell, Susan M. (      smitche1)   neutral good     Fighter( 10)     9740
      Mittal, Sandeep K. (      smitta1)    chaotic good      Cleric(  2)     2706


Sorted by alignment:
        Tuveson, Eric A. (      etuves1)    chaotic evil       Thief(  7)     9245
       Seo, Maximilian (          mseo1)    chaotic evil      Ranger(  3)     3205
          Joshua, Aimee (       ajoshu1)    chaotic evil       Thief(  4)     3733
            Fu, David E. (         dfu1)    chaotic good      Ranger( 12)    10011
      Palewicz, David E. (      dpalew1)    chaotic good       Thief(  3)      765
      Mittal, Sandeep K. (      smitta1)    chaotic good      Cleric(  2)     2706
      Orticke, Alecia G. (      aortic1)    chaotic good      Cleric(  4)     2048
       Cloud, Clinton E. (      ccloud2)    chaotic good    Magician(  9)     9583
        Frankle, Alan E. (     frankle1)    chaotic good     Fighter(  9)     8924
   Macdonald, Matthew S. (     matmacd1)    chaotic good     Fighter(  5)     3797
      Nguyen, Michael D. (     mnguyen1)    chaotic good      Ranger(  9)     9490
      Szrom, Michelle L. (      mszrom1)    chaotic good      Cleric(  9)     9660
    Chandlee, Richard C. (      rchand1)     lawful evil      Ranger(  3)     3196
         Moore, Brian A. (      bmoore1)     lawful evil      Ranger(  4)     4016
   Rusinko, Nicholas D. (       nrusin1)     lawful good       Thief(  5)     5095
           Raby, Adam M. (       araby1)  lawful neutral    Magician(  9)     8814
      Mcelvaney, Erin L. (      emcelv1)  lawful neutral      Cleric( 11)     9896
         Cobb, Milton T. (         cobb)  lawful neutral     Fighter(  9)     9169
          Allen, Kevin P. (     kallen1)    neutral good     Paladin(  6)     6721
      Mitchell, Susan M. (      smitche1)   neutral good     Fighter( 10)     9740
        Roberts, Eric J. (      erober3)    neutral good     Fighter(  7)     7765
       Shenvi, Shilpa P. (      shenvi1)    neutral good       Thief( 10)     9573
       Baker, Matthew R. (      mbaker5)    true neutral       Thief(  4)     4313
       Helton, Robert L. (      helton1)    true neutral     Paladin(  3)     3034
      Napier, Matthew A. (      mnapie1)    true neutral    Magician(  5)     4539
River[15]%

River[15]% exit
exit
```

```c
/* Files qsort3.c

   Demonstrating use of callback functions in qsort().
   This time we sort an array of pointers.
*/
#include <stdio.h>
#include <stdlib.h>
#include <string.h>


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

typedef struct {
    char   realname[32] ;
    char   nickname[16] ;
    char   alignment[20] ;
    char   role[20] ;
    int    points ;
    int    level ;

} Player ;


Player records[25] = {
    {"Allen, Kevin P.", "kallen1", "neutral good", "Paladin", 6721, 6},
    {"Baker, Matthew B.", "mbaker5", "true neutral", "Thief", 4313, 6},
    {"Chandlee, Richard C.", "rchand1", "lawful evil", "Ranger", 3196, 3},
    {"Cloud, Clinton E.", "ccloud2", "chaotic good", "Magician", 9583, 9},
    {"Cobb, Milton T.", "cobb", "lawful neutral", "Fighter", 9169, 9},
    {"Frankle, Alan E.", "frankle1", "chaotic good", "Fighter", 8924, 9},
    {"Fu, David E.", "dfu1", "chaotic good", "Ranger", 10011, 12},
    {"Helton, Robert L.", "helton1", "true neutral", "Paladin", 3034, 3},
    {"Joshua, Almee", "ajoshu1", "chaotic evil", "Thief", 3733, 4},
    {"Macdonald, Matthew S.", "matmand1", "chaotic good", "Fighter", 3797, 5},
    {"Moelvaney, Erin L.", "emoelv1", "lawful neutral", "Cleric", 9896, 11},
    {"Mitchell, Susan M.", "smitche1", "neutral good", "Fighter", 9740, 10},
    {"Mittal, Sandeep K.", "smittal", "chaotic good", "Cleric", 2706, 2},
    {"Moore, Brian A.", "bmoore1", "lawful evil", "Ranger", 4016, 4},
    {"Napier, Matthew A.", "mnapier1", "true neutral", "Magician", 4539, 5},
    {"Nguyen, Michael D.", "mnguyen1", "chaotic good", "Ranger", 9490, 9},
    {"Ortiche, Alecia G.", "aortic1", "chaotic good", "Cleric", 2048, 4},
    {"Palewicz, David E.", "dpalew1", "chaotic good", "Thief", 765, 3},
    {"Raby, Adam M.", "araby1", "lawful neutral", "Magician", 8814, 9},
    {"Roberts, Eric J.", "erober3", "neutral good", "Fighter", 7765, 7},
    {"Rusinko, Nicholas D.", "nrusin1", "lawful good", "Thief", 5095, 5},
    {"Seo, Maximilian", "mseo1", "chaotic evil", "Ranger", 3205, 3},
    {"Shanvi, Shilpa P.", "shanvi1", "neutral good", "Thief", 9573, 10},
    {"Szrom, Michelle L.", "mszrom1", "chaotic good", "Cleric", 9660, 9},
    {"Toveson, Eric A.", "etovea1", "chaotic evil", "Thief", 9245, 7}
} ;


void PrintPlayer(Player *p) {

    printf("%12s (%16s) %20s %20s%8d%8d\n",
        p->realname, p->nickname, p->alignment, p->role, p->level,
        p->points) ;
}
```

```c
int byPoints(Player **p1, Player **p2) {
   return (*p1)->points - (*p2)->points ;
}

int byNickname(Player **p1, Player **p2) {
   return strcmp((*p1)->nickname, (*p2)->nickname) ;
}

int byAlignment(Player **p1, Player **p2) {
   return strcmp((*p1)->alignment, (*p2)->alignment) ;
}

int main () {

   int i ;
   Player *rec_ptrs[25] ;

   for (i = 0 ; i < 25 ; i++) {
      rec_ptrs[i] = &records[i] ;
   }


   printf("Original list:\n") ;
   for (i = 0 ; i < 25 ; i++) {
      PrintPlayer(rec_ptrs[i]) ;
   }


   qsort(rec_ptrs, 25, sizeof(Player *), (COMP_FUNC_PTR) &byPoints) ;

   printf("\n\nSorted by points:\n") ;
   for (i = 0 ; i < 25 ; i++) {
      PrintPlayer(rec_ptrs[i]) ;
   }


   qsort(rec_ptrs, 25, sizeof(Player *), (COMP_FUNC_PTR) &byNickname) ;

   printf("\n\nSorted by nickname:\n") ;
   for (i = 0 ; i < 25 ; i++) {
      PrintPlayer(rec_ptrs[i]) ;
   }


   qsort(rec_ptrs, 25, sizeof(Player *), (COMP_FUNC_PTR) &byAlignment) ;

   printf("\n\nSorted by alignment:\n") ;
   for (i = 0 ; i < 25 ; i++) {
      PrintPlayer(rec_ptrs[i]) ;
   }

   printf("\n\nOriginal list (again):\n") ;
   for (i = 0 ; i < 25 ; i++) {
      PrintPlayer(&records[i]) ;
   }

   return 0 ;
}
```

```
Script started on Wed Oct 17 23:15:52 2012
River[16]% gcc -Wall qsort3.c
River[17]%
River[17]% ./a.out
Original list:
        Allen, Kevin P. (     kallen1)     neutral good     Paladin(  6)     6721
      Baker, Matthew B. (    mbaker5)      true neutral       Thief(  6)     4313
   Chandlee, Richard C. (     rchand1)      lawful evil      Ranger(  3)     3196
       Cloud, Clinton E. (     ccloud2)    chaotic good    Magician(  9)     9583
        Cobb, Milton T. (        cobb)   lawful neutral     Fighter(  9)     9169
     Frankle, Alan E. (    frankle1)    chaotic good     Fighter(  9)     8924
          Fu, David E. (       dfu1)    chaotic good      Ranger( 12)    10011
     Helton, Robert L. (     helton1)      true neutral     Paladin(  3)     3034
       Joshua, Almee (     ajoshu1)    chaotic evil       Thief(  4)     3733
  Macdonald, Matthew S. (    matmand1)    chaotic good     Fighter(  5)     3797
    Moelvaney, Erin L. (     emoelv1)   lawful neutral      Cleric( 11)     9896
    Mitchell, Susan M. (    smitche1)    neutral good     Fighter( 10)     9740
      Mittal, Sandeep K. (     smittal)    chaotic good      Cleric(  2)     2706
       Moore, Brian A. (     bmoore1)     lawful evil      Ranger(  4)     4016
     Napier, Matthew A. (    mnapier1)      true neutral    Magician(  5)     4539
    Nguyen, Michael D. (    mnguyen1)    chaotic good      Ranger(  9)     9490
     Ortiche, Alecia G. (     aortic1)    chaotic good      Cleric(  4)     2048
   Palewicz, David E. (     dpalew1)    chaotic good       Thief(  3)      765
       Raby, Adam M. (       araby1)   lawful neutral    Magician(  9)     8814
    Roberts, Eric J. (     erober3)    neutral good     Fighter(  7)     7765
   Rusinko, Nicholas D. (     nrusin1)     lawful good       Thief(  5)     5095
        Seo, Maximilian (        mseo1)    chaotic evil      Ranger(  3)     3205
     Shanvi, Shilpa P. (    shanvi1)    neutral good       Thief( 10)     9573
     Szrom, Michelle L. (     mszrom1)    chaotic good      Cleric(  9)     9660
     Toveson, Eric A. (     etovea1)    chaotic evil       Thief(  7)     9245


Sorted by points:
   Palewicz, David E. (     dpalew1)    chaotic good       Thief(  3)      765
     Ortiche, Alecia G. (     aortic1)    chaotic good      Cleric(  4)     2048
     Mittal, Sandeep K. (     smittal)    chaotic good      Cleric(  2)     2706
     Helton, Robert L. (     helton1)      true neutral     Paladin(  3)     3034
   Chandlee, Richard C. (     rchand1)      lawful evil      Ranger(  3)     3196
        Seo, Maximilian (        mseo1)    chaotic evil      Ranger(  3)     3205
       Joshua, Almee (     ajoshu1)    chaotic evil       Thief(  4)     3733
  Macdonald, Matthew S. (    matmand1)    chaotic good     Fighter(  5)     3797
     Szrom, Michelle L. (     mszrom1)    chaotic good      Cleric(  9)     9660
     Baker, Matthew B. (    mbaker5)      true neutral       Thief(  6)     4313
     Napier, Matthew A. (    mnapier1)      true neutral    Magician(  5)     4539
   Rusinko, Nicholas D. (     nrusin1)     lawful good       Thief(  5)     5095
       Allen, Kevin P. (     kallen1)     neutral good     Paladin(  6)     6721
     Roberts, Eric J. (     erober3)    neutral good     Fighter(  7)     7765
       Raby, Adam M. (       araby1)   lawful neutral    Magician(  9)     8814
     Frankle, Alan E. (    frankle1)    chaotic good     Fighter(  9)     8924
        Cobb, Milton T. (        cobb)   lawful neutral     Fighter(  9)     9169
     Toveson, Eric A. (     etovea1)    chaotic evil       Thief(  7)     9245
    Nguyen, Michael D. (    mnguyen1)    chaotic good      Ranger(  9)     9490
     Shanvi, Shilpa P. (    shanvi1)    neutral good       Thief( 10)     9573
       Cloud, Clinton E. (     ccloud2)    chaotic good    Magician(  9)     9583
     Szrom, Michelle L. (     mszrom1)    chaotic good      Cleric(  9)     9660
    Mitchell, Susan M. (    smitche1)    neutral good     Fighter( 10)     9740
    Moelvaney, Erin L. (     emoelv1)   lawful neutral      Cleric( 11)     9896
          Fu, David E. (       dfu1)    chaotic good      Ranger( 12)    10011
```

```
Sorted by nickname:
                 Joshua, Aimee (          ajoshu1)      chaotic evil              Thief(  4)        3733
            Orticke, Alecia G. (          aortic1)      chaotic good             Cleric(  4)        2048
                 Raby, Adam M. (           araby1)    lawful neutral         Magician(  9)        8814
               Moore, Brian A. (          bmoore1)       lawful evil             Ranger(  4)        4016
             Cloud, Clinton E. (          ccloud2)      chaotic good           Magician(  9)        9583
               Cobb, Milton T. (             cobb)    lawful neutral            Fighter(  9)        9169
                  Fu, David E. (             dfu1)      chaotic good             Ranger( 12)       10011
            Palewicz, David E. (          dpalew1)      chaotic good              Thief(  3)         765
            Mcelvaney, Erin L. (          emcelv1)    lawful neutral             Cleric( 11)        9896
              Roberts, Eric J. (          erober3)      neutral good            Fighter(  7)        7765
              Tuveson, Eric A. (          etuves1)      chaotic evil              Thief(  7)        9245
              Frankle, Alan E. (         frankle1)      chaotic good            Fighter(  9)        8924
             Helton, Robert L. (          helton1)      true neutral            Paladin(  3)        3034
               Allen, Kevin P. (          kallen1)      neutral good            Paladin(  6)        6721
         Macdonald, Matthew S. (         matmacd1)      chaotic good            Fighter(  5)        3797
             Baker, Matthew R. (          mbaker5)      true neutral              Thief(  4)        4313
            Napier, Matthew A. (          mnapie1)      true neutral           Magician(  5)        4539
            Nguyen, Michael D. (         mnguyen1)      chaotic good             Ranger(  9)        9490
               Seo, Maximilian (            mseo1)      chaotic evil             Ranger(  3)        3205
            Szrom, Michelle L. (          mszrom1)      chaotic good             Cleric(  9)        9660
          Rusinko, Nicholas D. (          nrusin1)       lawful good              Thief(  5)        5095
          Chandlee, Richard C. (          rchand1)       lawful evil             Ranger(  3)        3196
             Shenvi, Shilpa P. (          shenvi1)      neutral good              Thief( 10)        9573
            Mitchell, Susan M. (          smitchel)     neutral good            Fighter( 10)        9740
            Mittal, Sandeep K. (          smittal)      chaotic good             Cleric(  2)        2706

Sorted by alignment:
              Tuveson, Eric A. (          etuves1)      chaotic evil              Thief(  7)        9245
               Seo, Maximilian (            mseo1)      chaotic evil             Ranger(  3)        3205
                 Joshua, Aimee (          ajoshu1)      chaotic evil              Thief(  4)        3733
                  Fu, David E. (             dfu1)      chaotic good             Ranger( 12)       10011
            Palewicz, David E. (          dpalew1)      chaotic good              Thief(  3)         765
            Mittal, Sandeep K. (          smittal)      chaotic good             Cleric(  2)        2706
            Orticke, Alecia G. (          aortic1)      chaotic good             Cleric(  4)        2048
             Cloud, Clinton E. (          ccloud2)      chaotic good           Magician(  9)        9583
              Frankle, Alan E. (         frankle1)      chaotic good            Fighter(  9)        8924
         Macdonald, Matthew S. (         matmacd1)      chaotic good            Fighter(  5)        3797
            Nguyen, Michael D. (         mnguyen1)      chaotic good             Ranger(  9)        9490
            Szrom, Michelle L. (          mszrom1)      chaotic good             Cleric(  9)        9660
          Chandlee, Richard C. (          rchand1)       lawful evil             Ranger(  3)        3196
               Moore, Brian A. (          bmoore1)       lawful evil             Ranger(  4)        4016
          Rusinko, Nicholas D. (          nrusin1)       lawful good              Thief(  5)        5095
                 Raby, Adam M. (           araby1)    lawful neutral           Magician(  9)        8814
            Mcelvaney, Erin L. (          emcelv1)    lawful neutral             Cleric( 11)        9896
               Cobb, Milton T. (             cobb)    lawful neutral            Fighter(  9)        9169
               Allen, Kevin P. (          kallen1)      neutral good            Paladin(  6)        6721
            Mitchell, Susan M. (          smitchel)     neutral good            Fighter( 10)        9740
              Roberts, Eric J. (          erober3)      neutral good            Fighter(  7)        7765
             Shenvi, Shilpa P. (          shenvi1)      neutral good              Thief( 10)        9573
             Baker, Matthew R. (          mbaker5)      true neutral              Thief(  4)        4313
             Helton, Robert L. (          helton1)      true neutral            Paladin(  3)        3034
            Napier, Matthew A. (          mnapie1)      true neutral           Magician(  5)        4539
```

```
Original list (again):
               Allen, Kevin P. (          kallen1)      neutral good            Paladin(  6)        6721
             Baker, Matthew R. (          mbaker5)      true neutral              Thief(  4)        4313
          Chandlee, Richard C. (          rchand1)       lawful evil             Ranger(  3)        3196
             Cloud, Clinton E. (          ccloud2)      chaotic good           Magician(  9)        9583
               Cobb, Milton T. (             cobb)    lawful neutral            Fighter(  9)        9169
              Frankle, Alan E. (         frankle1)      chaotic good            Fighter(  9)        8924
                  Fu, David E. (             dfu1)      chaotic good             Ranger( 12)       10011
             Helton, Robert L. (          helton1)      true neutral            Paladin(  3)        3034
                 Joshua, Aimee (          ajoshu1)      chaotic evil              Thief(  4)        3733
         Macdonald, Matthew S. (         matmacd1)      chaotic good            Fighter(  5)        3797
            Mcelvaney, Erin L. (          emcelv1)    lawful neutral             Cleric( 11)        9896
            Mitchell, Susan M. (          smitchel)     neutral good            Fighter( 10)        9740
            Mittal, Sandeep K. (          smittal)      chaotic good             Cleric(  2)        2706
               Moore, Brian A. (          bmoore1)       lawful evil             Ranger(  4)        4016
            Napier, Matthew A. (          mnapie1)      true neutral           Magician(  5)        4539
            Nguyen, Michael D. (         mnguyen1)      chaotic good             Ranger(  9)        9490
            Orticke, Alecia G. (          aortic1)      chaotic good             Cleric(  4)        2048
            Palewicz, David E. (          dpalew1)      chaotic good              Thief(  3)         765
                 Raby, Adam M. (           araby1)    lawful neutral           Magician(  9)        8814
              Roberts, Eric J. (          erober3)      neutral good            Fighter(  7)        7765
          Rusinko, Nicholas D. (          nrusin1)       lawful good              Thief(  5)        5095
               Seo, Maximilian (            mseo1)      chaotic evil             Ranger(  3)        3205
             Shenvi, Shilpa P. (          shenvi1)      neutral good              Thief( 10)        9573
            Szrom, Michelle L. (          mszrom1)      chaotic good             Cleric(  9)        9660
              Tuveson, Eric A. (          etuves1)      chaotic evil              Thief(  7)        9245
River[18]%
River[18]% exit
exit

Script done on Wed Oct 17 23:16:03 2012
```