```
/* qsort example */
#include <stdio.h>
#include <stdlib.h>

typedef struct tableEntry tableType;

struct tableEntry {
  int num;
  char *wrd;
};

tableType *table;
int tableSize;

void readTable()
{
int i,j,k;
char buffer[81];

scanf("%d",&tableSize);
table=(tableType*) malloc(tableSize*sizeof(tableType));
if (!table)
{
  printf("malloc fail\n");
  exit(0);
}
for (i=0;i<tableSize;i++)
{
  scanf("%d %s",&(table[i].num),buffer);
  k=strlen(buffer);
  table[i].wrd=malloc(k+1);
  if (!(table[i].wrd))
  {
    printf("malloc fail\n");
    exit(0);
  }
  for (j=0;j<=k;j++)
    table[i].wrd[j]=buffer[j];
}
}

void printTable()
{
int i;

for (i=0;i<tableSize;i++)
  printf("%8d %s\n",table[i].num,table[i].wrd);
}

int numCompare(x,y)
tableType *x,*y;
{
return x->num - y->num;
}

void sortByNumber()
{
qsort((void*) table,tableSize,sizeof(tableType),numCompare);
}

int strCompare(x,y)
tableType *x,*y;
{
return strcmp(x->wrd,y->wrd);
}

void sortByWord()
{
qsort((void*) table,tableSize,sizeof(tableType),strCompare);
}

int revStrCompare(x,y)
tableType *x,*y;
{
return strcmp(y->wrd,x->wrd);
```

```
}

void reverseSortByWord()
{
qsort((void*) table,tableSize,sizeof(tableType),revStrCompare);
}

main()
{
readTable();
printf("Input:\n");
printTable();
sortByNumber();
printf("Sorted by number:\n");
printTable();
sortByWord();
printf("Sorted by word:\n");
printTable();
reverseSortByWord();
printf("Reverse sorted by word:\n");
printTable();
}
[weems@reptar~/docs/NOTES2320:96] cc qsortExample.c
using gcc
[weems@reptar~/docs/NOTES2320:97] cat veryshort.in
5
300dog
212cat
443pizza
867coke
342asparagus
[weems@reptar~/docs/NOTES2320:98] a.out < veryshort.in
Input:
     300 dog
     212 cat
     443 pizza
     867 coke
     342 asparagus
Sorted by number:
     212 cat
     300 dog
     342 asparagus
     443 pizza
     867 coke
Sorted by word:
     342 asparagus
     212 cat
     867 coke
     300 dog
     443 pizza
Reverse sorted by word:
     443 pizza
     300 dog
     867 coke
     212 cat
     342 asparagus
// Finds the kth element in an unsorted table by qsort (slow)
// or using quicksort-style partitioning (usually fast)
#include <stdio.h>
#include <stdlib.h>
#include <sys/time.h>
#include <sys/resource.h>

float elapsedCPU()
{
struct rusage rusage;
getrusage(RUSAGE_SELF,&rusage);
return rusage.ru_utime.tv_sec+rusage.ru_utime.tv_usec/1000000.0
  + rusage.ru_stime.tv_sec+rusage.ru_stime.tv_usec/1000000.0;
}

int generateRandom(minRange,maxRange)
int minRange,maxRange;
{
/* returns integer in range minRange <= x <= maxRange*/
```

```
return minRange+abs(random()) % (maxRange-minRange+1);
}

void generateRandomTable(int *arr,int n,int maxrange,int seed)
{
int i;

srandom(seed);
for (i=0;i<n;i++)
  arr[i]=generateRandom(0,maxrange);
}

int partition(int *arr,int p,int r)
// From CLRS, 2nd ed.
{
int x,i,j,temp;

x=arr[r];
i=p-1;
for (j=p;j<r;j++)
  if (arr[j]<=x)
  {
    i++;
    temp=arr[i];
    arr[i]=arr[j];
    arr[j]=temp;
  }
temp=arr[i+1];
arr[i+1]=arr[r];
arr[r]=temp;
return i+1;
}

int quickSelection(int *arr,int n,int k)
{
int lowRange,highRange,q;

lowRange=0;
highRange=n-1;
while (lowRange<=highRange)
{
  //printf("quickSelection subscript range %d..%d\n",lowRange,highRange);
  q=partition(arr,lowRange,highRange);
  if (k-1==q)
    return q;
  else if (k-1<q)
    highRange=q-1;
  else
    lowRange=q+1;
}
printf("quickSelection error\n");
exit(0);
}

int intCompare(const void* xin, const void* yin)
{
int *x,*y;

x=(int*) xin;
y=(int*) yin;
return (*x) - (*y);
}

main()
{
int i;
int seed,n,k,maxrange;
int *a;
float startCPU;

printf("enter seed, n, k, maxrange: ");
fflush(stdout);
scanf("%d %d %d %d",&seed,&n,&k,&maxrange);
a=(int*) malloc(n*sizeof(int));
if (a==NULL)
```

```
  exit(0);

generateRandomTable(a,n,maxrange,seed);

startCPU=elapsedCPU();
printf("quickSelection:  The value with rank %d is %d\n",
  k,a[quickSelection(a,n,k)]);
printf("Time used is %f\n", elapsedCPU()-startCPU);

generateRandomTable(a,n,maxrange,seed);

startCPU=elapsedCPU();
qsort(a,n,sizeof(int),intCompare);
printf("qsort: The value with rank %d is %d\n",k,a[k-1]);
printf("Time used is %f\n", elapsedCPU()-startCPU);
}
[weems@reptar~/docs/NOTES2320:79] cc selection.c
using gcc
[weems@reptar~/docs/NOTES2320:83] a.out
enter seed, n, k, maxrange: 222 500000 10000 999999999
quickSelection:  The value with rank 10000 is 14155713
Time used is 0.192276
qsort: The value with rank 10000 is 14155713
Time used is 3.060904
[weems@reptar~/docs/NOTES2320:84] a.out
enter seed, n, k, maxrange: 222 1000000 10000 999999999
quickSelection:  The value with rank 10000 is 7145757
Time used is 0.896754
qsort: The value with rank 10000 is 7145757
Time used is 6.467149
[weems@reptar~/docs/NOTES2320:85] a.out
enter seed, n, k, maxrange: 222 2000000 10000 999999999
quickSelection:  The value with rank 10000 is 3612010
Time used is 1.230060
qsort: The value with rank 10000 is 3612010
Time used is 13.526408
[weems@reptar~/docs/NOTES2320:86] a.out
enter seed, n, k, maxrange: 222 4000000 10000 999999999
quickSelection:  The value with rank 10000 is 1771686
Time used is 2.088253
qsort: The value with rank 10000 is 1771686
Time used is 28.909500
[weems@reptar~/docs/NOTES2320:87] a.out
enter seed, n, k, maxrange: 222 8000000 10000 999999999
quickSelection:  The value with rank 10000 is 890823
Time used is 3.897135
qsort: The value with rank 10000 is 890823
Time used is 60.005630
[weems@reptar~/docs/NOTES2320:88] a.out
enter seed, n, k, maxrange: 222 16000000 10000 999999999
quickSelection:  The value with rank 10000 is 446893
Time used is 15.119547
qsort: The value with rank 10000 is 446893
Time used is 124.468903
```