Example of Quicksort using Hoarse Partition and Random Partition Algorithms

Posted by Alejandro G. Carlstein Ramos Mejia on October 15, 2010 October 15, 2010 About Programming / Algorithms / ANSI/POSIX C

Example of Quicksort using Hoarse partition and random partition algorithms.

NOTIFICATION: These examples are provided for educational purposes. Using this code is under your own responsibility and risk. The code is given 'as is'. I do not take responsibilities of how they are used.

quicksort.c:

```
/* Framework for the sorting programs.
 * Program: 04
 * Author: Alejandro G. Carlstein
 * Description: This program will implement quicksort
 * (using the pseudocode from the book), applying
 * the Hoare partition function, and randomization.
 * This program should run with data where all the values
 * are equal, and also when the data is in sorted
 * (or reverse sorted) order.
 */

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

#define MAX_SIZE 1000000

#define DBG_LV1 0
#define DBG_LV2 0

void Quicksort(int array[], int p, int r);
int Hoare_partition(int array[], int p, int r);
int Randomized_partition(int array[], int p, int r);
void Exchange(int *a, int *b);
int Get_random(int min, int max);

int data[MAX_SIZE];

int n;

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

 int i;

 int n;
```

```
 int m = 0;

 int begin = 1;

 int ia_keys[MAX_SIZE];

 srand (time (NULL));

 /* Read in the data */
 n = 0;
    while (scanf('%d', &data[n]) == 1)
     ++n;

 /* Print ou t the data */
 if (DBG_LV2)
  for (i = 0; i < n; ++i)
      printf('[%d] %d\n', i, data[i]);

 Quicksort(data, 0, n);

 /* Print out the data */
 for (i = 0; i < n; ++i)
     if (DBG_LV2){
      printf('[%d] %d\n', i, data[i]);
     }else{
      printf('%d\n', data[i]);
     }//end if

 if (DBG_LV2)
  printf('+[10]: %d \n', data[10]);

 return 0;
}

/*
 * Quicksort
 */
void Quicksort(int array[], int p, int r){

 if (DBG_LV1)
  printf('Quicksort(p: %d, r: %d)\n', p, r);

 int q;

 if (p < r){

  q = Randomized_partition(array, p, r);

  Quicksort(array, p, q);

  Quicksort(array, q + 1, r);
```

```
 }// end if

}

/*
 * Hoare partition Book
 */
int Hoare_partition(int array[], int p, int r){

 if (DBG_LV1)printf('Hoare_partition_book(p: %d, r:%d)\n', p, r);

 int b_done = 1;

 // pivot value
 int x = array[p];

 int i = p - 1;

 int j = r + 1;

 if (DBG_LV2)
  printf('i: %d, j: %d\n', i, j);

 while (b_done){

  do{
   --j;
  }while (array[j] < x);

  do{
   ++i;
  }while (array[i] > x);

  if (DBG_LV2)
   printf('+i: %d, j: %d\n', i, j);

  if (i < j){

   Exchange(&array[i], &array[j]);

  }else{

   b_done = 0;

  }//end if

 }// end while

 return j;
}

/*
 * Randomized partition
```

```
 */
int Randomized_partition(int array[], int p, int r){

 if (DBG_LV1)
  printf('Randomized_partition(p: %d, r: %d)\n', p, r);

 int i = Get_random(p, r);

 Exchange(&array[i], &array[r]);

 return Hoare_partition(array, p, r);

}

/*
 * Exchange swap the values holded by two variables
 */
void Exchange(int *a, int *b){

 if (DBG_LV1)
  printf('Exchange(value a: %d, value b: %d)\n', *a, *b);

  int temp;

  temp = *a;
  *a = *b;
  *b = temp;
}

/*
 * get random number
 */
int Get_random(int min, int max){

 if (DBG_LV1)
  printf('Get_random(min: %d, max: %d)\n', min, max);

 int rtn = 0;

 if (max > min){

  rtn = (rand() % (max - min)) + min;

 }else{

  fprintf(stderr, '[X] ERROR: minimum value cannot exceed maximun value\n');

  exit(1);

 }//end if

 return rtn;
}
```

If you encounter any problems or errors, please let me know by providing an example of the code, input, output, and an explanation. Thanks.