```
************************************************************
Report: hw4
Author: F74046022 陳冠仁 <jeremy851004@gmail.com>
Class: 乙班
Description:
How do you finish this homework?
隨機產生(整數/浮點數)存入陣列
使用 quicksort 排列(升冪)
將陣列的前後對調(降冪)

What did you learned from this homework?
quicksort 的邏輯和如何使用
qsort 對不同種類變數的使用方法
************************************************************
Code:
#include <stdio.h>
#include <stdlib.h>
#include <time.h>

int i,j;

int main(int argc, char *argv[])
{
	int n=atoi(argv[1]), x=atoi(argv[2]);
	int a[n], da[n];
	float f[n], df[n];

	if(!(x==0 || x==1) || n<1)
	{
		printf("\nINVALID INPUT\n\n\n");
		return 0;
	}

	srand(time(0));

	if(x==0)
	{
		for(i=0; i<n; i++)
```

```
                a[i]=rand()%10000;

            printf("\n\nBEFORE SORTING : \n");
            print_array_int(a, n);

            quicksort_int(a, 0, n-1);

            for(i=0, j=n-1; i<n; i++, j--)
                da[j]=a[i];
            //sort in descending order

            printf("\n\nAFTER SORTING : \n");
            print_array_int(da, n);
        }

        if(x==1)
        {
            for(i=0; i<n; i++)
                f[i]=rand()/(RAND_MAX/10000.0);

            printf("\n\nBEFORE SORTING : \n");
            print_array_float(f, n);

            quicksort_float(f, 0, n-1);

            for(i=0, j=n-1; i<n; i++, j--)
                df[j]=f[i];
            //sort in descending order

            printf("\n\nAFTER SORTING : \n");
            print_array_float(df, n);
        }

        printf("\n\n\n");
        return 0;
}

int print_array_int(int a[],int n)
```

```
{
	for(i=0, j=0; i<n; i++, j++)
	{
		if(j/10>(j-1)/10)
			printf("\n");
		printf("%5d", a[i]);
	}
}

int print_array_float(float f[],int n)
{
	for(i=0, j=0; i<n; i++, j++)
	{
		if(j/10>(j-1)/10)
			printf("\n");
		printf("%8.2f", f[i]);
	}
}

int quicksort_int(int a[], int low, int high)
{
	int middle;

	if(low>=high) return;
	middle=split_int(a, low, high);
	quicksort_int(a, low, middle-1);
	quicksort_int(a, middle+1, high);
}

int split_int(int a[], int low, int high)
{
	int part_element=a[low];

	for(;;)
	{
		while(low<high && part_element<=a[high])
			high--;
		if(low>=high) break;
```

```
            a[low++]=a[high];

            while(low<high && a[low]<=part_element)
                  low++;
            if(low>=high) break;
            a[high--]=a[low];
      }

      a[high]=part_element;
      return high;
}

int quicksort_float(float a[], int low, int high)
{
      float middle;

      if(low>=high) return;
      middle=split_float(a, low, high);
      quicksort_float(a, low, middle-1);
      quicksort_float(a, middle+1, high);
}

int split_float(float a[], int low, int high)
{
      float part_element=a[low];

      for(;;)
      {
            while(low<high && part_element<=a[high])
                  high--;
            if(low>=high) break;
            a[low++]=a[high];

            while(low<high && a[low]<=part_element)
                  low++;
            if(low>=high) break;
            a[high--]=a[low];
      }
```

```
        a[high]=part_element;
        return high;
}
```

Compilation:

```
        gcc -o hw4 hw4.c
```

Execution:

```
        ./hw4 (N) (0 or 1)
```

Output:

```
BEFORE SORTING :
 7909 3325 3351 2716 1234 7248  493 8728 7060 9853
 9676   67 2153 3933 8550

AFTER SORTING :
 9853 9676 8728 8550 7909 7248 7060 3933 3351 3325
 2716 2153 1234  493   67


BEFORE SORTING :
 7052.64 3295.44 9658.54 2971.50 8864.85  566.84 2617.91
5983.69 7143.21 8300.68
 1766.37 6149.43  213.08 5851.28 3155.43

AFTER SORTING :
 9658.54 8864.85 8300.68 7143.21 7052.64 6149.43 5983.69
5851.28 3295.44 3155.43
 2971.50 2617.91 1766.37  566.84  213.08
```