SEARCHING AND SORTING

Insertion Sort - Part 1

#include <stdio.h>

void print(int ar\_size, int\* ar) { int i;

for(i=0; i<ar\_size; i++) { printf("%d ", ar[i]);

printf("\n");

#include <string.h> #include <math.h> #include <stdlib.h> #include <assert.h>

/\* Head ends here \*/

void insertionSort(int ar\_size, int \* ar) { int j = ar\_size-1;

int v = ar[j];

while(v < ar[j-1]) {

ar[j] = ar[j-1];

--›

print(ar\_size, ar);



print(ar\_size, ar);

#### /\* Tail starts here \*/ int main() (

int \_ar\_size; scanf("%d", &\_ar\_size); int \_ar[\_ar\_size], \_ar\_i;

for(\_ar\_i = 0; \_ar\_i < \_ar\_size; \_ar\_i++) ( scanf("%d", &\_ar[\_ar\_i]);

insertionSort(\_ar\_size, \_ar);

return 0;

## Insertion Sort - Part 2

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

/\* Head ends here \*/

void insertionSort(int ar\_size, int \* ar) ( for (int i = 1; i < ar\_size; ++i) (

int j = i - 1;

int p = ar[i];

while (j >= 0 && p < ar[j]) {

ar[j+1] = ar[j];

ar[j+1] = p;

printf("%d", ar[0]);

for (int k = 1; k < ar\_size; ++k) (

printf(" %d", ar[k]);

printf("\n");

/\* Tail starts here \*/ int main() {

int \_ar\_size; scanf("%d", &\_ar\_size);

int \_ar[\_ar\_size], \_ar\_i;

for(\_ar\_i = 0; \_ar\_i < \_ar\_size; \_ar\_i++) ( scanf("%d", &\_ar[\_ar\_i]);

insertionSort(\_ar\_size, \_ar);

return 0;

## Correctness and the Loop Invariant

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

/\* Head ends here \*/ #include <stddef.h>

void insertionSort(int ar\_size, int \* ar) {



int value;

for(i=1;i<ar\_size;i++)

value=ar[i];

j=i-1;

while(j>=0 && value<ar[j])

ar[j+1]=ar[j];

j=j-1;

ar[j+1]=value;

for(j=0;j<ar\_size;j++)

#### printf("%d",ar[j]);

printf(" ");

/\* Tail starts here \*/ int main(void) {

int \_ar\_size; scanf("%d", &\_ar\_size); int \_ar[\_ar\_size], \_ar\_i;

for(\_ar\_i = 0; \_ar\_i < \_ar\_size; \_ar\_i++) { scanf("%d", &\_ar[\_ar\_i]);

insertionSort(\_ar\_size, \_ar);

return 0;

# Running Time of Algorithms

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

/\* Head ends here \*/

void insertionSort(int ar\_size, int \* ar,int \*shifts) ( int temp=ar[ar\_size-1],i;

for(i=ar\_size-2;i>=0;i--)

if(ar[i]>temp)(

ar[i+1]=ar[i];

\*shifts=\*shifts+1;

#### else

break;

ar[i+1]=temp;

/\* Tail starts here \*/

int main() (

int \_ar\_size,i,j,shifts=0; scanf("%d", &\_ar\_size); int \_ar[\_ar\_size], \_ar\_i;

for(\_ar\_i = 0; \_ar\_i < \_ar\_size; \_ar\_i++) { scanf("%d", &\_ar[\_ar\_i]);

for(i=2;i<=\_ar\_size;i++)

insertionSort(i, \_ar,&shifts);

printf("%d",shifts); return 0;

### Counting Sort 1

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

int main() (

#### int n,i;

int b[100],a;

scanf("%d",&n);

for(i=0;i<100;i++)

b[i]=0;

for(i=0;i<n;i++)

scanf("%d",&a);

b[a]++;

for(i=0;i<100;i++)

printf("%d ", b[i]);

return 0;

# RECURSION AND BIT MANIPULATION

Crossword Puzzle

### The Power Sum

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

int the\_power\_sum(int n, int m,int p)( int tmp = pow(m,p);

if(tmp == n) return 1;

if(tmp > n) return 0;

return the\_power\_sum(n,m+1,p) + the\_power\_sum(n-tmp, m+1,p);

int main() ( int n,p;

scanf("%d%d",&n,&p);

printf("%d", the\_power\_sum(n,1,p));

return 0;

Counter Game #include <stdio.h> #include <string.h> #include <math.h> #include <stdlib.h>

int isPow2(long unsigned int);

unsigned long int IargePow(long unsigned int); int main() (

int t,i,win;

long unsigned int n; scanf("%d",&t);

for(i=0;i<t;++i)

win=0;

scanf("%lu",&n);

if(n==1)

printf("Richard\n");

else

while(n!=1)

if(isPow2(n))

n>>——1;

else

n-=largePow(n);

++win;

if(win%2==0)

printf("Richard\n");

else

printf("Louise\n");

return 0;

int isPow2(long unsigned int n) return !(n&(n-1));

long unsigned int IargePow(long unsigned int n)

long unsigned int m; while(n)

#### m=n;

n=n&(n-1);

return m;





