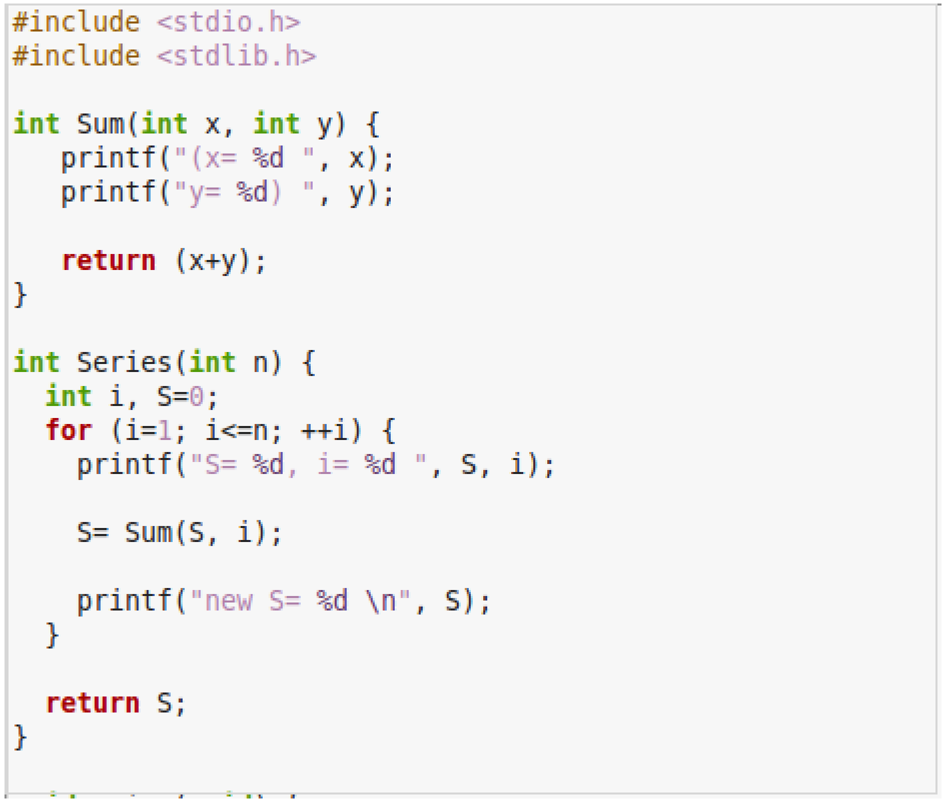
Recursion (1A)

Young Won Lim
3/20/15

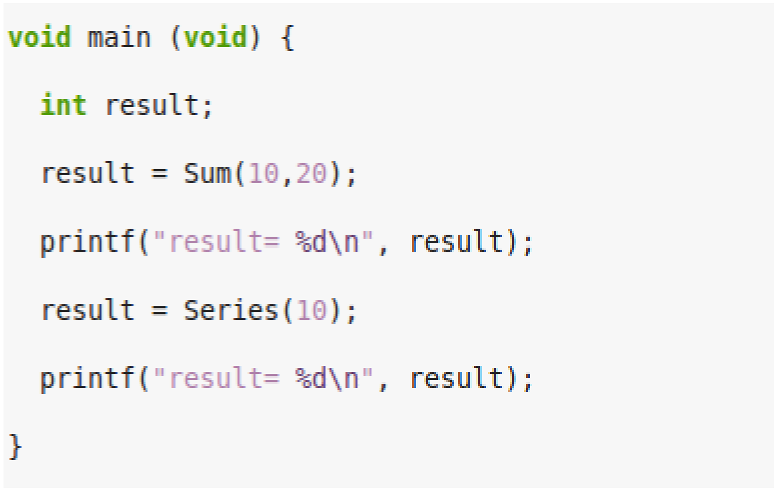
Please send corrections (or suggestions) to youngwlim@hotmail.com.

This document was produced by using OpenOffice and Octave.

Function Call

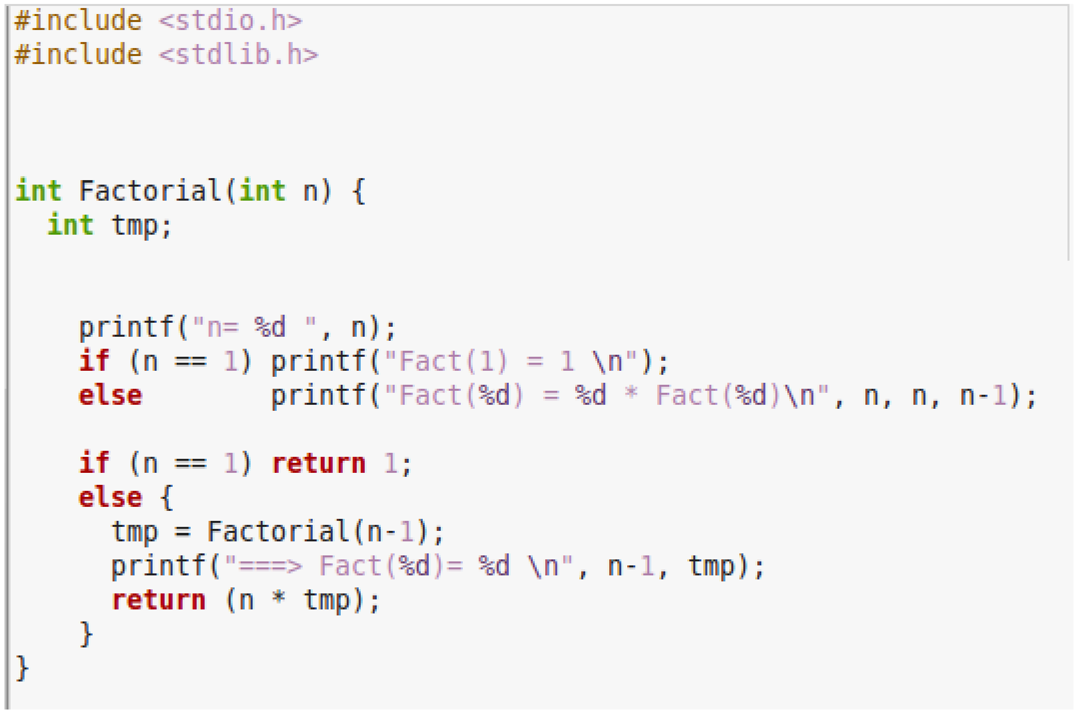


```c
#include <stdio.h>
#include <stdlib.h>

int Sum(int x, int y) {
   printf("(x= %d ", x);
   printf("y= %d) ", y);

   return (x+y);
}

int Series(int n) {
  int i, S=0;
  for (i=1; i<=n; ++i) {
    printf("S= %d, i= %d ", S, i);

    S= Sum(S, i);

    printf("new S= %d \n", S);
  }

  return S;
}
```

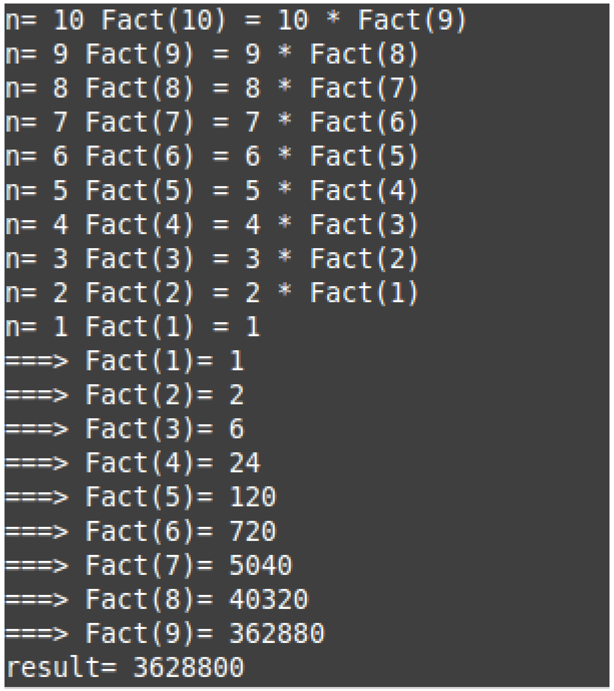


```c
void main (void) {

  int result;

  result = Sum(10,20);

  printf("result= %d\n", result);

  result = Series(10);

  printf("result= %d\n", result);

}
```

Recursive Function – Factorial

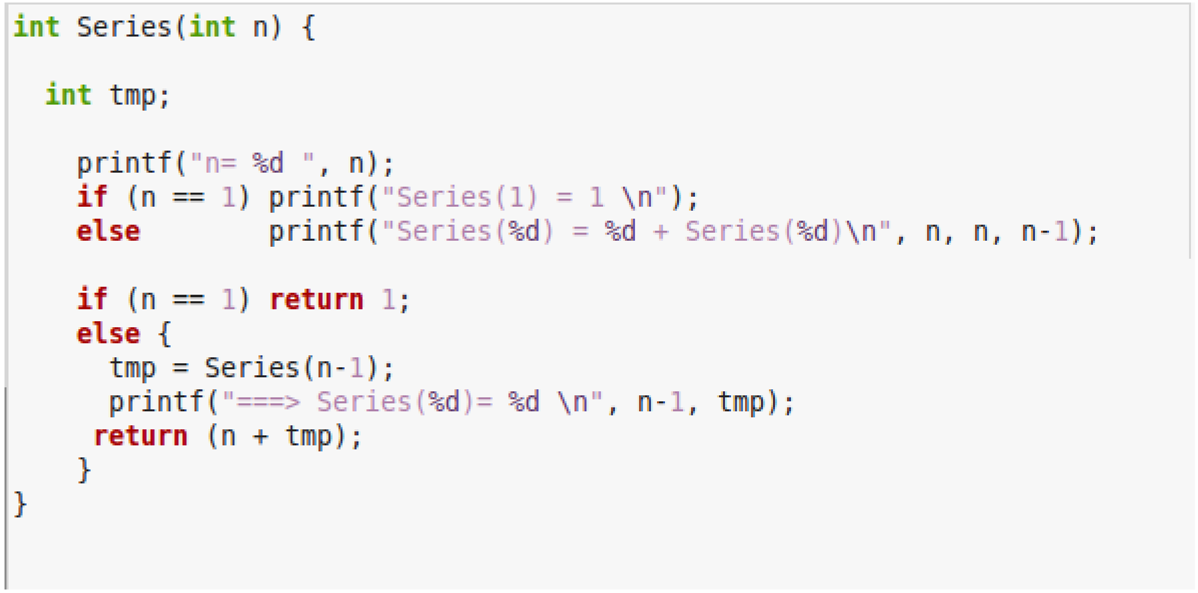


```c
#include <stdio.h>
#include <stdlib.h>



int Factorial(int n) {
  int tmp;


    printf("n= %d ", n);
    if (n == 1) printf("Fact(1) = 1 \n");
    else        printf("Fact(%d) = %d * Fact(%d)\n", n, n, n-1);

    if (n == 1) return 1;
    else {
      tmp = Factorial(n-1);
      printf("===> Fact(%d)= %d \n", n-1, tmp);
      return (n * tmp);
    }
}
```

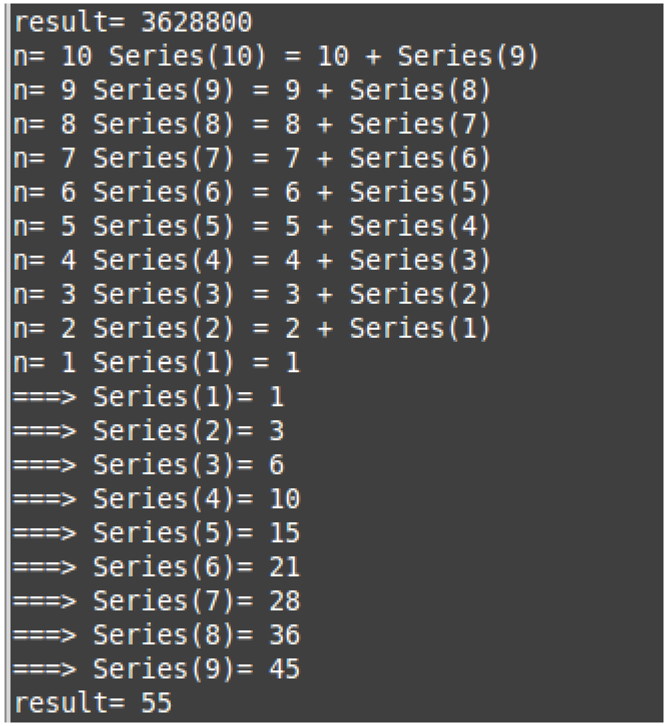


```
n= 10 Fact(10) = 10 * Fact(9)
n= 9 Fact(9) = 9 * Fact(8)
n= 8 Fact(8) = 8 * Fact(7)
n= 7 Fact(7) = 7 * Fact(6)
n= 6 Fact(6) = 6 * Fact(5)
n= 5 Fact(5) = 5 * Fact(4)
n= 4 Fact(4) = 4 * Fact(3)
n= 3 Fact(3) = 3 * Fact(2)
n= 2 Fact(2) = 2 * Fact(1)
n= 1 Fact(1) = 1
===> Fact(1)= 1
===> Fact(2)= 2
===> Fact(3)= 6
===> Fact(4)= 24
===> Fact(5)= 120
===> Fact(6)= 720
===> Fact(7)= 5040
===> Fact(8)= 40320
===> Fact(9)= 362880
result= 3628800
```

Recursive Function – Series

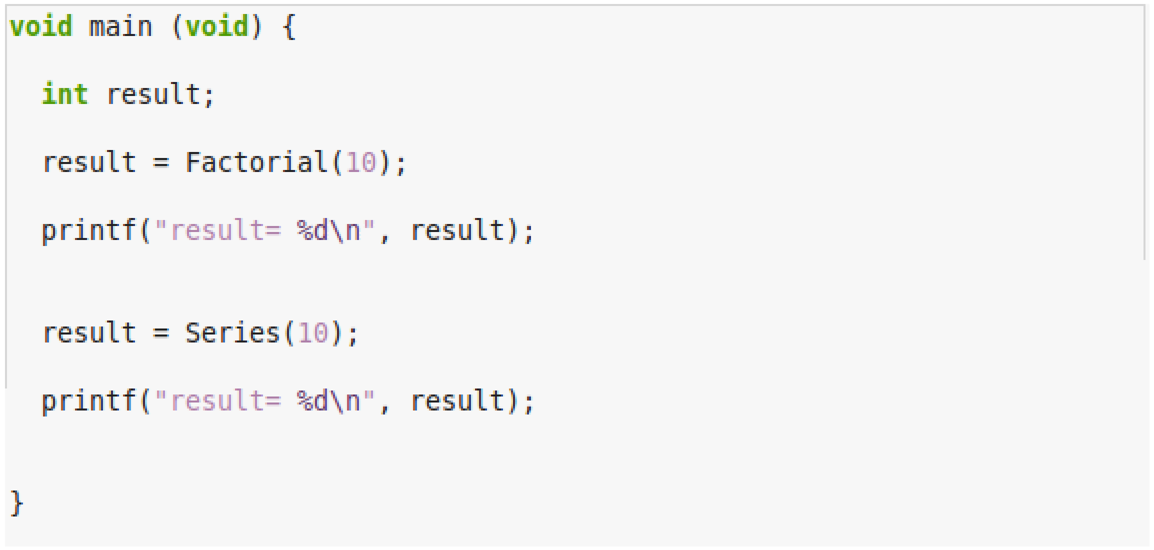


```
int Series(int n) {

  int tmp;

    printf("n= %d ", n);
    if (n == 1) printf("Series(1) = 1 \n");
    else        printf("Series(%d) = %d + Series(%d)\n", n, n, n-1);

    if (n == 1) return 1;
    else {
      tmp = Series(n-1);
      printf("===> Series(%d)= %d \n", n-1, tmp);
     return (n + tmp);
    }
}
```



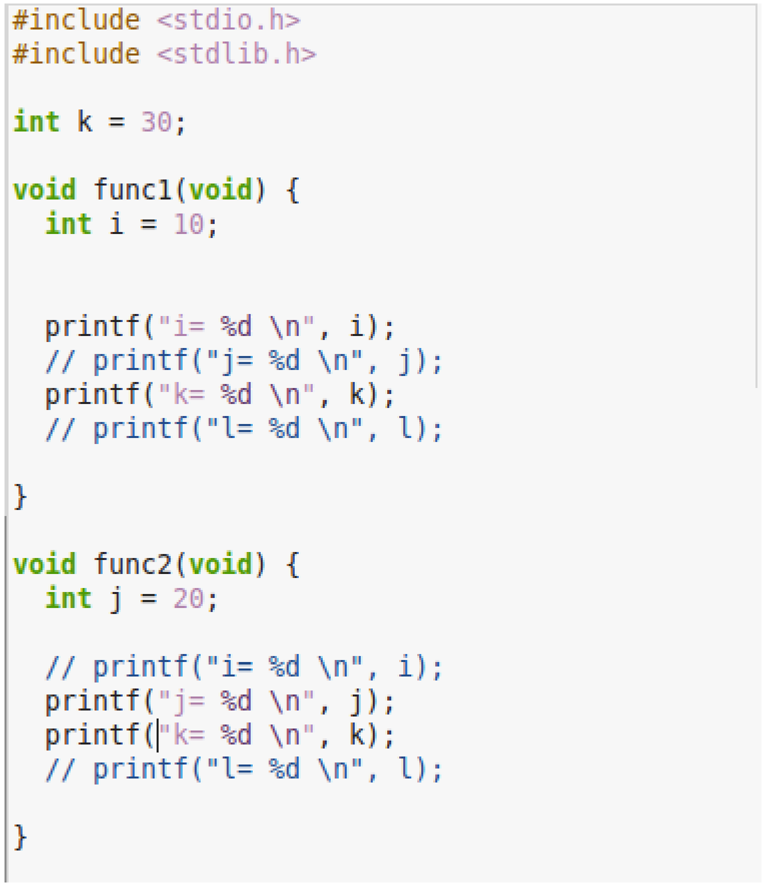
```
result= 3628800
n= 10 Series(10) = 10 + Series(9)
n= 9 Series(9) = 9 + Series(8)
n= 8 Series(8) = 8 + Series(7)
n= 7 Series(7) = 7 + Series(6)
n= 6 Series(6) = 6 + Series(5)
n= 5 Series(5) = 5 + Series(4)
n= 4 Series(4) = 4 + Series(3)
n= 3 Series(3) = 3 + Series(2)
n= 2 Series(2) = 2 + Series(1)
n= 1 Series(1) = 1
===> Series(1)= 1
===> Series(2)= 3
===> Series(3)= 6
===> Series(4)= 10
===> Series(5)= 15
===> Series(6)= 21
===> Series(7)= 28
===> Series(8)= 36
===> Series(9)= 45
result= 55
```

Recursive Function – main

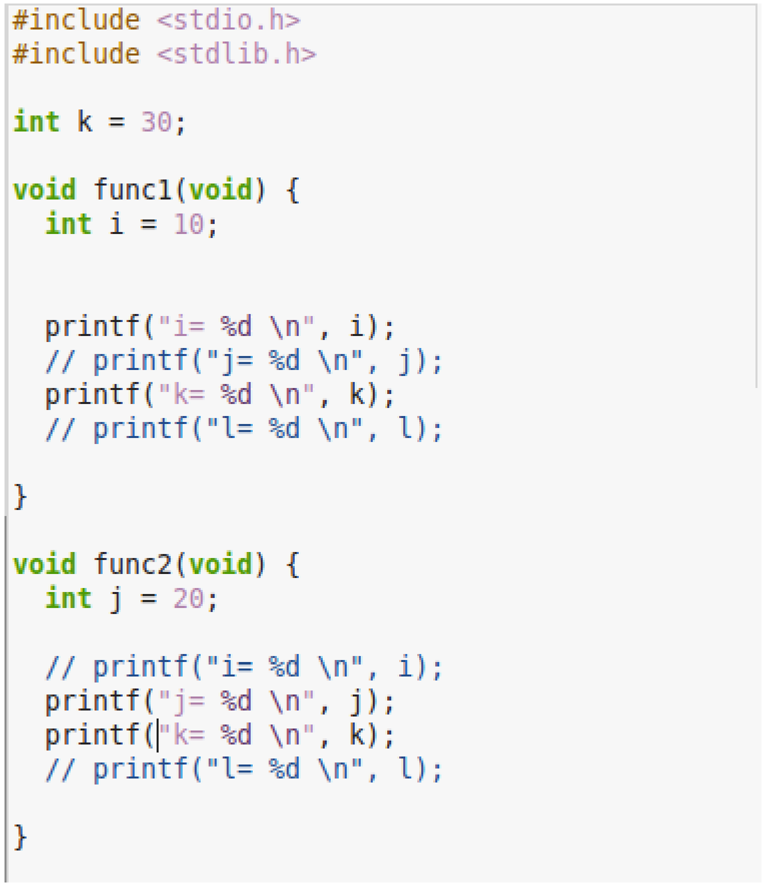


```
void main (void) {

  int result;

  result = Factorial(10);

  printf("result= %d\n", result);


  result = Series(10);

  printf("result= %d\n", result);


}
```

Variable Scope



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

int k = 30;

void func1(void) {
  int i = 10;


  printf("i= %d \n", i);
  // printf("j= %d \n", j);
  printf("k= %d \n", k);
  // printf("l= %d \n", l);

}

void func2(void) {
  int j = 20;

  // printf("i= %d \n", i);
  printf("j= %d \n", j);
  printf("k= %d \n", k);
  // printf("l= %d \n", l);

}
```

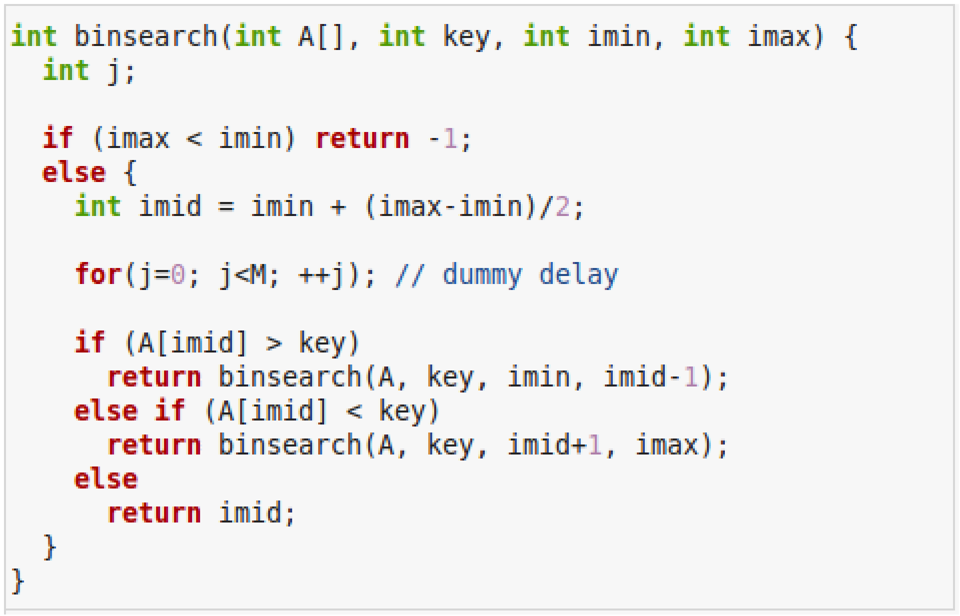


```c
#include <stdio.h>
#include <stdlib.h>

int k = 30;

void func1(void) {
  int i = 10;


  printf("i= %d \n", i);
  // printf("j= %d \n", j);
  printf("k= %d \n", k);
  // printf("l= %d \n", l);

}

void func2(void) {
  int j = 20;

  // printf("i= %d \n", i);
  printf("j= %d \n", j);
  printf("k= %d \n", k);
  // printf("l= %d \n", l);

}
```

Linear Search

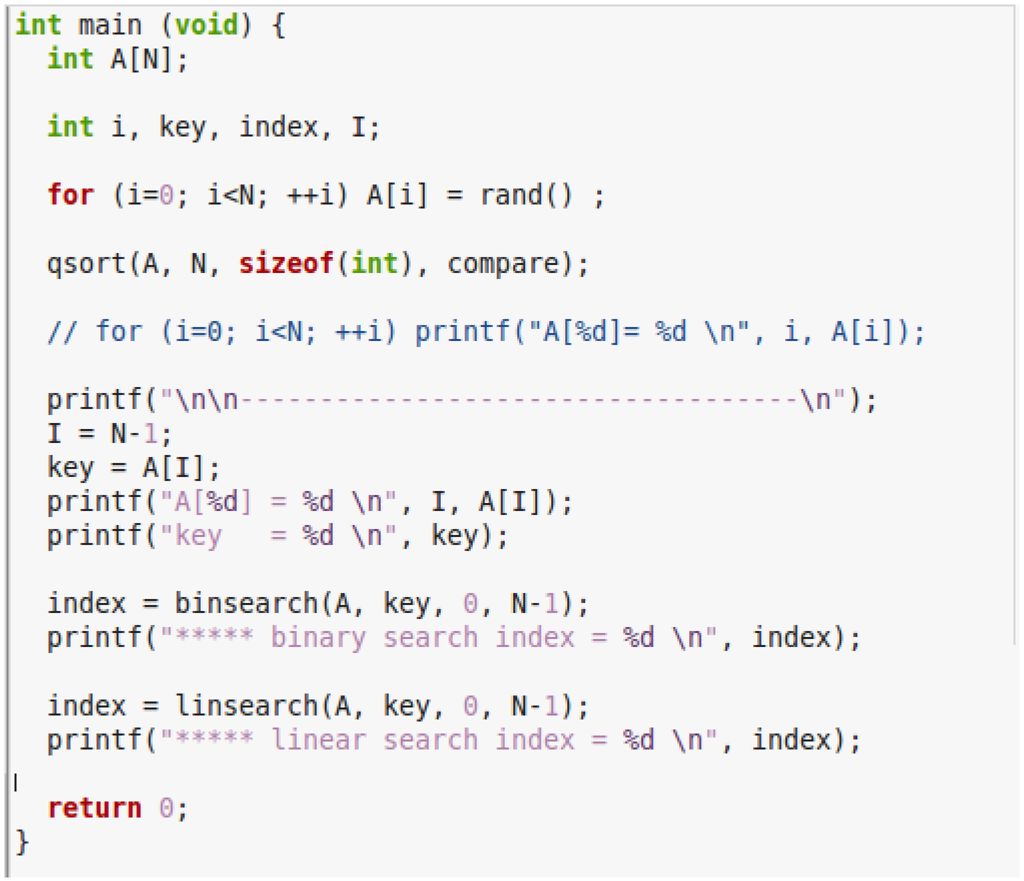
```
#include <stdio.h>
#include <stdlib.h>
#define N 1000000
#define M 1000

int linsearch(int A[], int key, int imin, int imax) {
  int i, j;

  for (i=0; i<N; ++i) {
    for(j=0; j<M; ++j); // dummy delay
    if (A[i] == key) return i;
  }

  return -1;
}
```

Binary Search



```c
int binsearch(int A[], int key, int imin, int imax) {
  int j;

  if (imax < imin) return -1;
  else {
    int imid = imin + (imax-imin)/2;

    for(j=0; j<M; ++j); // dummy delay

    if (A[imid] > key)
      return binsearch(A, key, imin, imid-1);
    else if (A[imid] < key)
      return binsearch(A, key, imid+1, imax);
    else
      return imid;
  }
}
```

Random Number Generator



```
int main (void) {
  int A[N];

  int i, key, index, I;

  for (i=0; i<N; ++i) A[i] = rand() ;

  qsort(A, N, sizeof(int), compare);

  // for (i=0; i<N; ++i) printf("A[%d]= %d \n", i, A[i]);

  printf("\n\n----------------------------------\n");
  I = N-1;
  key = A[I];
  printf("A[%d] = %d \n", I, A[I]);
  printf("key   = %d \n", key);

  index = binsearch(A, key, 0, N-1);
  printf("***** binary search index = %d \n", index);

  index = linsearch(A, key, 0, N-1);
  printf("***** linear search index = %d \n", index);

  return 0;
}
```

References

[1] http://en.wikipedia.org/
[2]