```
# `6-recursion`

# 6-recursion

## Recursion

- `main-re.c`

- `min.c`
  - stack/heap
  - automatic variable
- `min-re.c`

- `sum-re.c`
  - static storage
  - static variable

- `fib-re.c`
- `fib-iter.c`

- `gcd-re.c`
- `gcd-iter.c`

- `bsearch-iter.c`
- `bsearch-re.c`

## Backup

- `hanoi.c`
- `quicksort.c`
```

```c
// file: main-re.c
//
// Created by hfwei on 2023/11/9.
//
// WARNING: You can even call the "main" function in itself.
// But, do NOT write code like this.
// Never call the "main" function in your own code.
//

#include <stdio.h>

int main(int argc, char *argv[]) {
  if (argc == 1) {
    return 0;
  }

  printf("%s\n", argv[argc - 1]);

  main(argc - 1, argv);

  return 0;
}
```

```c
//
// Created by hfwei on 2023/11/9.
// Visualization of function call: https://pythontutor.com/visualize.html#code=%23include%20%3Cstdio.h%3E%0A%0Aint%20Min%28int%20a,%20int%20b%29%3B%0A%0Aint%20main%28%29%20%7B%0A%20%20int%20a%20%3D%2025%3B%0A%20%20int%20b%20%3D%2037%3B%0A%20%20%0A%20%20int%20min%20%3D%20Min%28a,%20b%29%3B%0A%20%20printf%28%22%25d%22,%20min%29%3B%0A%0A%20%20return%200%3B%0A%7D%0A%0Aint%20Min%28int%20a,%20int%20b%29%20%7B%0A%20%20return%20a%20%3E%20b%20%3F%20b%20%3A%20a%3B%0A%7D&cumulative=false&heapPrimitives=nevernest&mode=edit&origin=opt-frontend.js&py=c_gcc9.3.0&rawInputLstJSON=%5B%5D&textReferences=false
//

#include <stdio.h>

int Min(int a, int b);

int main() {
  int a = 25;
  int b = 37;

  int min = Min(a, b);
  printf("%d", min);

  return 0;
}

int Min(int a, int b) {
  return a > b ? b : a;
}
```

```c
//
// Created by hfwei on 2023/11/9.
// Visualization: https://pythontutor.com/visualize.html#code=%23include%20%3Cstdio.h%3E%0A%0A%23define%20NUM%203%0Aint%20numbers%5BNUM%5D%20%3D%20%7B65,%2028,%2037%7D%3B%0A%0Aint%20Min%28const%20int%20nums%5B%5D,%20int%20len%29%3B%0A%0Aint%20main%28%29%20%7B%0A%20%20int%20min%20%3D%20Min%28numbers,%20NUM%29%3B%0A%20%20%0A%20%20printf%28%22min%20%3D%20%25d%5Cn%22,%20min%29%3B%0A%0A%20%20return%200%3B%0A%7D%0A%0Aint%20Min%28const%20int%20numbers%5B%5D,%20int%20len%29%20%7B%0A%20%20if%20%28len%20%3D%3D%201%29%20%7B%0A%20%20%20%20return%20numbers%5B0%5D%3B%0A%20%20%7D%0A%0A%20%20int%20partial_min%20%3D%20Min%28numbers,%20len%20-%201%29%3B%0A%20%20return%20numbers%5Blen%20-%201%5D%20%3C%20partial_min%20%3F%20numbers%5Blen%20-%201%5D%20%3A%20partial_min%3B%0A%7D&cumulative=true&heapPrimitives=nevernest&mode=edit&origin=opt-frontend.js&py=c_gcc9.3.0&rawInputLstJSON=%5B%5D&textReferences=false
//

#include <stdio.h>

#define NUM 3
const int numbers[NUM] = { 65, 28, 37 };

int Min(const int nums[], int len);

int main() {
  int min = Min(numbers, NUM);

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

  return 0;
}

// compute the minimum of nums[0 .. len - 1]
int Min(const int nums[], int len) {
  if (len == 1) {
    return nums[0];
  }

  int partial_min = Min(nums, len - 1);

  return partial_min < nums[len - 1] ? partial_min : nums[len - 1];
}
```

```c
//
// Created by hfwei on 2023/11/9.
//
// Visualization: https://pythontutor.com/visualize.html#code=%23include%20%3Cstdio.h%3E%0A%0Aint%20Sum%28int%20numbers%5B%5D,%20int%20len%29%3B%0A%0Aint%20main%28%29%20%7B%0A%20%20%20%20int%20numbers%5B%5D%20%3D%20%7B1,%202,%203,%204,%205%7D%3B%0A%0A%20%20%20%20int%20sum%20%3D%20Sum%28numbers,%20sizeof%20numbers%20/%20sizeof%20numbers%5B0%5D%29%3B%0A%20%20%20%20printf%28%22sum%20%3D%20%25d%5Cn%22,%20sum%29%3B%0A%0A%20%20%20%20return%200%3B%0A%7D%0A%0Aint%20Sum%28int%20numbers%5B%5D,%20int%20len%29%20%7B%0A%20%20%20%20if%20%28len%20%3D%3D%200%29%20%7B%0A%20%20%20%20%20%20%20%20return%200%3B%0A%20%20%20%20%7D%0A%0A%20%20%20%20int%20partial_sum%20%3D%20Sum%28numbers,%20len%20-%201%29%3B%0A%0A%20%20%20%20int%20sum%20%3D%20numbers%5Blen%20-%201%5D%20%2B%20partial_sum%3B%0A%0A%20%20%20%20return%20sum%3B%0A%7D&cumulative=false&heapPrimitives=nevernest&mode=edit&origin=opt-frontend.js&py=c_gcc9.3.0&rawInputLstJSON=%5B%5D&textReferences=false
//

#include <stdio.h>

int Sum(const int nums[], int len);

int main() {
  const int numbers[] = { 1, 2, 3, 4, 5 };

  // sizeof: operator
  int sum = Sum(numbers, sizeof numbers / sizeof numbers[0]);
  printf("sum = %d\n", sum);

  return 0;
}

int Sum(const int nums[], int len) {
  if (len == 0) {
    return 0;
  }

  int partial_sum = Sum(nums, len - 1);

  return partial_sum + nums[len - 1];
}
```

```c
//
// Visualization (for n = 4): https://pythontutor.com/render.html#code=%23include%20%3Cstdio.h%3E%0A%0Along%20long%20Fib%28int%20n%29%3B%0A%0Aint%20main%28%29%20%7B%0A%20%20int%20n%20%3D%204%3B%0A%0A%20%20printf%28%22%25lld%5Cn%22,%20Fib%28n%29%29%3B%0A%7D%0A%0Along%20long%20Fib%28int%20n%29%20%7B%0A%20%20if%20%28n%20%3D%3D%200%29%20%7B%0A%20%20%20%20return%200%3B%0A%20%20%7D%0A%0A%20%20if%20%28n%20%3D%3D%201%29%20%7B%0A%20%20%20%20return%201%3B%0A%20%20%7D%0A%0A%20%20return%20Fib%28n%20-%201%29%20%2B%20Fib%28n%20-%202%29%3B%0A%7D&cumulative=false&curInstr=55&heapPrimitives=nevernest&mode=display&origin=opt-frontend.js&py=c_gcc9.3.0&rawInputLstJSON=%5B%5D&textReferences=false
// Created by hfwei on 2023/11/9.
//

#include <stdio.h>

long long Fib(int n);

int main() {
  int n;
  scanf("%d", &n);

  printf("%lld\n", Fib(n));

  return 0;
}

long long Fib(int n) {
  if (n <= 1) {
    return n;
  }

  return Fib(n - 1) + Fib(n - 2);
}
```

```c
//
// Created by hfwei on 2023/11/9.
//

#include <stdio.h>

// Fib(92) =  7 540 113 804 746 346 429
// long long: 9 223 372 036 854 775 807
// Fib(93) = 12 200 160 415 121 876 738
int main() {
  int n;
  scanf("%d", &n);

  long long fib0 = 0;
  long long fib1 = 1;

  long long fib2 = 0;
  for (int i = 2; i <= n; i++) {
    fib2 = fib0 + fib1;

    fib0 = fib1;
    fib1 = fib2;
  }

  printf("Fib(%d) = %lld ", n, fib2);

  return 0;
}
```

```c
//
// Created by hfwei on 2023/11/9.
//

#include <stdio.h>

#define LEN 93

int main() {
  long long fibs[LEN] = { 0, 1 };

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

  for (int i = 2; i <= n; ++i) {
    fibs[i] = fibs[i - 1] + fibs[i - 2];
  }

  printf("Fib(%d) = %lld\n", n, fibs[n]);

  return 0;
}
```

```c
// file: gcd-re.c
//
//  Euclidean algorithm:
//  gcd(a, b) = gcd(b, a % b)
//
//  Visualization (gcd(64, 48) for illustration):
//    https://pythontutor.com/visualize.html#code=%23include%20%3Cstdio.h%3E%0A%0Aint%20GCD%28int%20a,%20int%20b%29%3B%0A%0Aint%20main%28%29%20%7B%0A%20%20int%20a%20%3D%2064%3B%0A%20%20int%20b%20%3D%2048%3B%0A%0A%20%20printf%28%22gcd%28%25d,%20%25d%29%20%3D%20%25d%5Cn%22,%20a,%20b,%20GCD%28a,%20b%29%29%3B%0A%0A%20%20return%200%3B%0A%7D%0A%0A//%20gcd%28130,%20124%29%20%3D%202%0A//%20gcd%28662,%20414%29%20%3D%202%0Aint%20GCD%28int%20a,%20int%20b%29%20%7B%0A%20%20if%20%28b%20%3D%3D%200%29%20%7B%0A%20%20%20%20return%20a%3B%0A%20%20%7D%0A%0A%20%20return%20GCD%28b,%20a%20%25%20b%29%3B%0A%7D&cumulative=true&heapPrimitives=nevernest&mode=edit&origin=opt-frontend.js&py=c_gcc9.3.0&rawInputLstJSON=%5B%5D&textReferences=false
//
// Created by hfwei on 2023/11/9.
//

#include <stdio.h>

int GCD(int a, int b);

int main() {
  int a = 0;
  int b = 0;
  scanf("%d %d", &a, &b);

  printf("GCD(%d, %d) = %d\n", a, b, GCD(a, b));

  return 0;
}

// gcd(130, 124) = 2
// gcd(414, 662) = 2
int GCD(int a, int b) {
  if (b == 0) {
    return a;
  }

  return GCD(b, a % b);
}
```

```c
// file: gcd-iter.c
//
// Euclidean algorithm:
// gcd(a, b) = gcd(b, a % b)
//
// Created by hfwei on 2023/11/9.

#include <stdio.h>

int GCD(int a, int b);

int main() {
  int a = 130;
  int b = 124;

  printf("gcd(%d, %d) = %d\n", a, b, GCD(a, b));

  return 0;
}

int GCD(int a, int b) {
  while (b != 0) {
    int tmp = b;
    b = a % b;
    a = tmp;
  }

  return a;
}
```

```c
//
// Created by hfwei on 2023/11/9.
//

#include <stdio.h>

#define LEN 10

// dictionary: out of any functions; global variables
// life time: program start to end
// scope: from this point on until the end of the file (file scope)
// int dictionary[LEN] = { 0, 1, 1, 2, 3, 5, 8, 13, 21, 34 };

/**
 * @brief Search for the key in the dict using the binary search algorithm.
 * @param key the key to search for
 * @param dict the dictionary to search
 * @param len the length of the dictionary
 * @return the index of the key in the dictionary; -1 if not found
 */
int BinarySearch(int key, const int dict[100], int len);

int main(void) {
  const int dictionary[LEN] = { 0, 1, 1, 2, 3, 5, 8, 13, 21, 34 };

  int key = 0;
  scanf("%d", &key);

  int index = BinarySearch(key, dictionary, LEN);

  if (index == -1) {
    printf("Not found!\n");
  } else {
    printf("The index of %d is %d.\n", key, index);
  }

  return 0;
}

int BinarySearch(int key, const int dict[], int len) {
  int low = 0;
  int high = len - 1;

  while (low <= high) {
    int mid = (low + high) / 2;

    if (key > dict[mid]) {
      low = mid + 1;
    } else if (key < dict[mid]) {
      high = mid - 1;
    } else {  // key == dict[mid]
      return mid;
```

```
53      }
54    }
55
56    return -1;
57  }
```

```c
// file: bsearch-re.c
//
// Visualization (search for 2 as an example):
//  https://pythontutor.com/visualize.html#code=%23include%20%3Cstdio.h%3E%0A%0A%23define%20LEN%2010%0A%0Aint%20BinarySearch%28int%20key,%20const%20int%20dict%5B%5D,%20int%20low,%20int%20high%29%3B%0A%0Aint%20main%28%29%20%7B%0A%20%20const%20int%20dictionary%5BLEN%5D%20%3D%20%7B%200,%201,%201,%202,%203,%205,%208,%2013,%2021,%2034%20%7D%3B%0A%0A%20%20int%20key%20%3D%202%3B%0A%0A%20%20printf%28%22The%20index%20of%20%25d%20is%20%25d.%5Cn%22,%20key,%0A%20%20%20%20%20%20%20%20%20%20BinarySearch%28key,%20dictionary,%200,%20LEN%20-%201%29%29%3B%0A%0A%20%20return%200%3B%0A%7D%0A%0Aint%20BinarySearch%28int%20key,%20const%20int%20dict%5B%5D,%20int%20low,%20int%20high%29%20%7B%0A%20%20if%20%28low%20%3E%20high%29%20%7B%0A%20%20%20%20return%20-1%3B%0A%20%20%7D%0A%0A%20%20int%20mid%20%3D%20%28low%20%2B%20high%29%20/%202%3B%0A%0A%20%20if%20%28dict%5Bmid%5D%20%3D%3D%20key%29%20%7B%0A%20%20%20%20return%20mid%3B%0A%20%20%7D%0A%0A%20%20if%20%28dict%5Bmid%5D%20%3E%20key%29%20%7B%0A%20%20%20%20return%20BinarySearch%28key,%20dict,%20low,%20mid%20-%201%29%3B%0A%20%20%7D%0A%0A%20%20return%20BinarySearch%28key,%20dict,%20mid%20%2B%201,%20high%29%3B%0A%7D&cumulative=true&heapPrimitives=nevernest&mode=edit&origin=opt-frontend.js&py=c_gcc9.3.0&rawInputLstJSON=%5B%5D&textReferences=false
// Created by hfwei on 2023/11/9.

#include <stdio.h>

#define LEN 10

int BinarySearch(int key, const int dict[], int low, int high);

int main() {
  const int dictionary[LEN] = { 0, 1, 1, 2, 3, 5, 8, 13, 21, 34 };

  int key;
  scanf("%d", &key);

  printf("The index of %d is %d.\n", key,
         BinarySearch(key, dictionary, 0, LEN - 1));

  return 0;
}

int BinarySearch(int key, const int dict[], int low, int high) {
//  if (low == high) {
//    if (dict[low] == key) {
//      return low;
//    }
//    return -1;
//  }

  if (low > high) {
    return -1;
  }
```

```
36
37   int mid = (low + high) / 2;
38
39   if (dict[mid] == key) {
40     return mid;
41   }
42
43   if (dict[mid] > key) {
44     return BinarySearch(key, dict, low, mid - 1);
45   }
46
47   return BinarySearch(key, dict, mid + 1, high);
48 }
```