CS1020E Tutorial + Lab 07

Mark NG

a0116298@u.nus.edu
http://mollymr305.github.io

October 14, 2016

Tutorial Solutions

"Tutorial 7 – Recursion"

Question 1: Simple Recursion

Write a recursive function gcd(int a, int b), which prints how
Euclid's algorithm is used to derive the greatest common divisor of
two integers a and b.

Question 1: Simple Recursion

Write a recursive function gcd(int a, int b), which prints how Euclid's algorithm is used to derive the greatest common divisor of two integers *a* and *b*.

$\star$ https://en.wikipedia.org/wiki/Euclidean_algorithm

Question 1: Simple Recursion

Write a recursive function gcd(int a, int b), which prints how Euclid's algorithm is used to derive the greatest common divisor of two integers *a* and *b*.

$\star$ https://en.wikipedia.org/wiki/Euclidean_algorithm

$$a = q_0 b + r_0$$

Question 1: Simple Recursion

Write a recursive function gcd(int a, int b), which prints how
Euclid's algorithm is used to derive the greatest common divisor of
two integers *a* and *b*.

$\star$ https://en.wikipedia.org/wiki/Euclidean_algorithm

$$a = q_0 b + r_0$$
$$b = q_1 r_0 + r_1$$

Question 1: Simple Recursion

Write a recursive function gcd(int a, int b), which prints how
Euclid's algorithm is used to derive the greatest common divisor of
two integers *a* and *b*.

$\star$ https://en.wikipedia.org/wiki/Euclidean_algorithm

$$a = q_0 b + r_0$$
$$b = q_1 r_0 + r_1$$
$$r_0 = q_2 r_1 + r_2$$

Question 1: Simple Recursion

Write a recursive function gcd(int a, int b), which prints how
Euclid's algorithm is used to derive the greatest common divisor of
two integers *a* and *b*.

$\star$ https://en.wikipedia.org/wiki/Euclidean_algorithm

$$a = q_0 b + r_0$$
$$b = q_1 r_0 + r_1$$
$$r_0 = q_2 r_1 + r_2$$
$$r_1 = q_3 r_2 + r_3$$

Question 1: Simple Recursion

Write a recursive function gcd(int a, int b), which prints how
Euclid's algorithm is used to derive the greatest common divisor of
two integers *a* and *b*.

⋆ https://en.wikipedia.org/wiki/Euclidean_algorithm

$$a = q_0 b + r_0$$
$$b = q_1 r_0 + r_1$$
$$r_0 = q_2 r_1 + r_2$$
$$r_1 = q_3 r_2 + r_3$$
$$\vdots$$

Question 1: Simple Recursion

The Solution.

```cpp
int gcd(int a, int b) {
    if (b == 0) return a;
    cout << "(" << a << ", " << b << ") " << a << " = "
        << (a/b) << " * " << b << " + " << (a%b) << endl;
    return gcd(b, a%b);
}
```
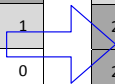
Question 2: Application of Recursion

- 2D integer array
- Each element is either 0 (white), 1 (light gray), 2 (dark gray).

```
void fill(int** arr, int numRows, int numCols,
        int currRow, int currCol, int newColour
        /*,one more param? */
        ) {
    // can use this method to recurse if necessary
}
void fill(int** arr, int numRows, int numCols,
        int startRow, int startCol, int newColour) {
    // client will use this method
}
```

Question 2: Application of Recursion

Question 2: Application of Recursion

Solution (Part 1)

```
void fill(int** arr, int nRows, int nCols,
          int currRow, int currCol,
          int newColo, int firstColo) {
    if (currRow < 0 || currRow >= nRows
        || currCol < 0 || currCol >= nCols
        || firstColo != arr[currRow][currCol])
        return;
    arr[currRow][currCol] = newColo;
    fill(arr,nRows,nCols,currRow-1,currCol,
        newColo,firstColo);
    fill(arr,nRows,nCols,currRow+1,currCol,
        newColo,firstColo);
    fill(arr,nRows,nCols,currRow,currCol-1,
        newColo,firstColo);
    fill(arr,nRows,nCols,currRow,currCol+1,
        newColo,firstColo);
}
```

Question 2: Application of Recursion

Solution (Part 2)

```
void fill(int** arr, int numRows, int numCols,
          int startRow, int startCol, int newColour) {
    int firstColour = arr[startRow][startCol];
    if (firstColour == newColour)
        return; // nothing to fill
    fill(arr, numRows, numCols,
        startRow, startCol, newColour, firstColour);
}
```

End of Tutorial Discussion

Note: Detailed solutions (i.e. the file T7_ans.pdf) will be released soon at

http://www.comp.nus.edu.sg/~stevenha/cs1020e.html

Take Home Lab

Some notes...

Take Home Lab

Some notes...

- **Important**: you must only use Queue and Stack for the first and second problem respectively.

Take Home Lab

Some notes...

- **Important**: you must only use Queue and Stack for the first and second problem respectively.
- First one should be easy. You should be able to complete it with only one queue.

Take Home Lab

Some notes...

- **Important**: you must only use Queue and Stack for the first and second problem respectively.
- First one should be easy. You should be able to complete it with only one queue.
- Second one is a bit tricky.

Take Home Lab

Some notes...

- **Important**: you must only use Queue and Stack for the first and second problem respectively.
- First one should be easy. You should be able to complete it with only one queue.
- Second one is a bit tricky. But you can also complete it similarly, with one stack.

Let's take a short break!

Recursion

Classical examples you should be aware of:

Recursion

Classical examples you should be aware of:

- Fibonacci.

Recursion

Classical examples you should be aware of:

- Fibonacci.
- Factorial.

Recursion

Classical examples you should be aware of:

- Fibonacci.

- Factorial.

- Binomial Coefficients.

Recursion

Classical examples you should be aware of:

- Fibonacci.

- Factorial.

- Binomial Coefficients. Hint: $\binom{n}{k} = \binom{n-1}{k-1} + \binom{n-1}{k}$.

Recursion

Classical examples you should be aware of:

- Fibonacci.
- Factorial.
- Binomial Coefficients. Hint: $\binom{n}{k} = \binom{n-1}{k-1} + \binom{n-1}{k}$.

Remark: Recursion is not the only way of computing these values. It is also not the best method.

Kattis Problem

https://open.kattis.com/problems/minimumscalar

Any Questions?

See you next week!