Department of Informatics
Database Technology Group
University of Zurich Solution Prof. Dr. Anton Dignös

Informatics II
Exercise 2 / **Solution**

Mar 08 , 2020

Recursion

Task 1

Solution: Let x be the base, and *pow* be the power. As $x^0 = 1$ if $x \neq 0$, the termination condition is $pow = 0$ and the output is $x^0 = 1$. The recursion is $x^{pow} = x * x^{pow-1}$. More specifically, assume the recursive function is `exponent(x, pow)`, then the result of `exponent(x, pow)` will be `base * exponent(x, pow - 1)`.

```
Algo: EXPONENT(x, pow)
if $pow == 0$ then
    return 1
else
    return $x$ * exponent(x, pow-1)
```

Task 2

Solution

1. First, the termination conditions are $n = 1$ and $n = 2$, and return 1 and 2 respectively.

2. Call `sequence(n-1)` and `sequence(n-2)` to calculate a_{n-1} and a_{n-2} respectively.

3. If a_{n-1} is divisible by 3, then $a_{n-1} = a_{n-1}/3$.

4. Finally, sum up a_{n-1} and a_{n-2} as a_n.

```
Algo: SEQUENCE(n)

if $n == 1$ then
    return 1
if $n == 2$ then
    return 2
else
    $a_{n-1}$ = sequence(n-1)
    $a_{n-2}$ = sequence(n-2)
    if $a_{n-1}\%3 == 0$ then
        $a_{n-1} = a_{n-1}$ / 3
    return $a_{n-1} + a_{n-2}$
call sequence(n)
```

Task 3

Solution The solution includes the iterative function and the recursive function.

- **Iterative function**. The iterative function includes three parts:
  - First the function calculates the length of the string (by using the `strlen` function in the previous exercise).
  - Then the function iterates from the first character until the last character '\0'. When iterating the string, the function checks if each character is uppercase. If the character is uppercase, then the function returns the current position and stops iterating.
  - If the function does not find any uppercase and reached '\0', then it should return -1 to indicate there is no uppercase character.

```
Algo: ITERATIVEFIRSTUPPER(str)

$i \leftarrow 0$
$len \leftarrow strlen(str)$
while $str[i] \neq$ '\0' do
    if 'A' $\leq str[i] \leq$ 'Z' then
        return $i$
    $i = i + 1$
return -1
```

- **Recursive function**. The recursive function takes two parameters: the string itself and current position, and traverses the string character by character.

  There are two termination conditions:

  - If the current character is '\0', which is the end of the input string. Return -1.
  - If the current character is an uppercase letter. Return current position.

  Then the recursive part of the function is `recursiveFirstUpper(str, pos)=recursiveFirstUpper(str, pos + 1)`. We start the recursive call with the parameter `pos=0`.

```
Algo: RECURSIVEFIRSTUPPER(str, pos)
if $str[pos]$ == '\0' then
    return -1
if 'A' ≤ $str[pos]$ ≤ 'Z' then
    return pos
return recursiveFirstUpper(str, pos+1)
```

Task 4

Solution There are the following termination conditions:

- If $i == j$, then the element is the last element at each row, and the function should return 1.
- $j == 0$, then the element is the first element at each row, and the function should return 1 as well.

The recursive part is `pascal(i, j) = pascal(i-1, j) + pascal(i-1, j-1)`.

```
Algo: PASCAL(i, j)
if $i == j$ then
    return 1
if $j == 0$ then
    return 1
return pascal(i - 1, j) + pascal(i - 1, j - 1)
```

```
Algo: PRINTPASCAL(n)
for $i = 0; i <= n; i++$ do
    printf("row %d: ", i);
    for $j = 0; j <= i; j++$ do
        printf("%d ", $pascal(i, j)$);
```

Note: The above algorithm only treats valid input, i.e. $i \leq j$. You are supposed to handle the invalid inputs in your own code.