Naseem Auguste

Computer Science I
Exercise: Recursion

We have gone through many examples in the class and those examples are available in the slides and uploaded codes. Try to test those codes and modify as you wish for getting more clarification.

In addition try the following:

1) What would be the output of the following recursive function if we call rec2(5) ?

```
void rec2(int x)
{
    if (x==0)
        return;
     rec2(x-1);
    printf("%d  ", x);
}
```

Answer: 1, 2, 3, 4, 5

2) Write a recursive function that calculates the sum $1^1 + 2^2 + 3^3 + \ldots + n^n$, given an integer value of n in between 1 and 9. You can write a separate power function in this process and call that power function as needed:

Power Function(if exp==0\return 1, Else return(base *Power(base, exp-1)),

```
int crazySum(int n);
```

if(n==1)return 1
else return pow(n,n)+ sum(n-1)),

3) Given the function below, what would the function call question3(10, 101) return?

```
int question3(int a, int b) {
        if (a == 0) return b;
        if (b == 0) return a;
        return question3(10*a+b%10, b/10);
}
```

Answer: 1010

4) Write a recursive function that converts a decimal number to an octal number.

int Oct[100], int i=0,
while (num != 0){
Oct[i] = n% 8,
n=n /8,
i++,}

5) The code below returns the number of zeros at the end of n! [factorial n]

```
int zeros(int n)
{
        int res = 0;
        while (n!=0)
        {
                res  += n/5;
                n /= 5;
        }
        return res;
}
```

Rewrite this method recursively:

```
if(n==0)return 0.
return n/5 + zeros(n/5);
```

6. Write a recursive function that returns the product of the digits of its integer input parameter, n. You omay assume that n is non-negative. For example, productDigits(243) should return 24, since 2 x 4 x 3 = 24.

```
int productDigits (int n) {
        while(n != 0)
          return n%10 * productDigits(n/10),
```

7. Let us define the weighted sum of an integer array a[0], a[1], a[2], …, a[n-1] be a[0]*1 + a[1]*2 + a[2]*3 + …+a[n-1]*n. For example, the weighted sum of the array [5,2,6] would be 5*1+2*2+6*3 = 27. Write a recursive function that takes in an array numbers and its length n, and returns its weighter sum. You can assume n is non-negative integer.

```
int weightedSum(int numbers[], int n) {
        if(n==0 ) return 0,
        else return num[n-1]* num[n-1] + weightedSum(num, n-1),
```

8. Mathematically, given a function f, we recursively define $f^k(n)$ as follows: if k = 1, $f^1(n) = f(n)$. Otherwise, for k > 1, $f^k(n) = f(f^{k-1}(n))$. Assume that there is an existing function f, which takes in a single integer and returns an integer. Write a ***recursive*** function fcomp, which takes in both n and k (k > 0), and returns $f^k(n)$.

```
int f(int n);
int fcomp(int n, int k){
```

```
if (k==1) return f(n);
return f(fcomp(n, k-1));
```

9. What would be the value of fun(7) for the following function?

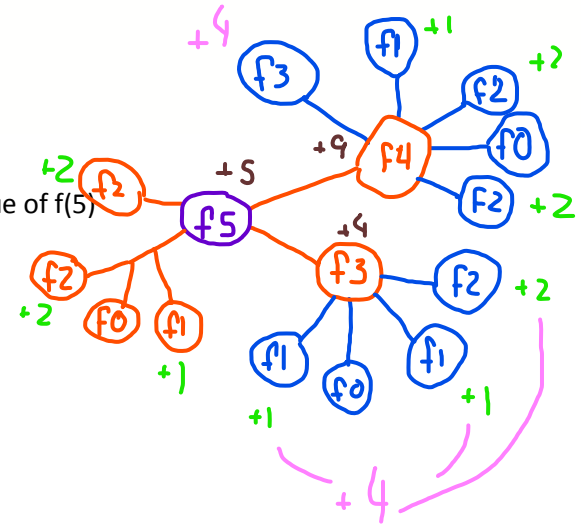
```
int fun(int x)
{
  if(x==0)
    return 0;
  if(x%3 ==0)
    return fun(x/3);
  return fun(x-1) + x;
}
```

9

10. Draw the recursion tree to find out the value of f(5)

```
int f(int n)
{
        int ans;
        int i;
        if(n<3)
                return n;
        ans = f(n/2);
        for(i=0; i<n; i++)
                ans += f(i);
        return ans;
}
```

9+4+5=18



11. Write a recursive function that returns 1 if an array of size n is in sorted order and 0 otherwise. Note: If array a stores 3, 6, 7, 7, 12, then isSorted(a, 5) should return 1 . If array b stores 3, 4, 9, 8, then isSorted(b, 4) should return 0.

```
int isSorted(int *array, int n){
```

```
if (n < 0) return 1;
if(array[n-1] < array[n-2]) return 0;
return isSorted(array, n-1);
```

11. Write recursive functions for inserting an item to the end of a linked list. Also, write another recursive function to delete a specific items from a linked list. ***// you don't have to submit this answer. But it would be good practice!***