Assignment 4 Part B

Sarvansh Prasher

Code:

1. (* Part A. Write a function sumList (L) that returns the sum of the numbers in the list L. *)

```
fun sumList [] = 0
|   sumList (head::tail) = head + sumList tail;
```

Sample program :

Query :

```
sumList [22,4,7,1];
```

Output :

```
val it = 34 : int
```

(* Part B. Write a function Fibonacci (n) that computes the nth Fibonacci number for a given number n. *)

```
fun fibonacci 0 = 1
|   fibonacci 1 = 1
|   fibonacci 2 = 1
|   fibonacci n = fibonacci(n-1) + fibonacci(n-2);
```

Sample program :

Query :

```
fibonacci 6;
```

Output :

```
val it = 8 : int
```

2. Reverse a List

```
fun rev r [] = r
```

```
| rev r ( head::tail) = rev(head::r) tail;
fun reverse list = rev [] list;
```

Sample program :

Query :

```
reverse([1, 2, 3, 4, 5]);
```

Output :

```
val it = [5,4,3,2,1] : int list
```

3. Rotate a list

```
fun recursiveRotate(L,H::T,num) =
 if num = 0 then
    [H] @T @ L
  else
   recursiveRotate(L@[H],T,num-1);

fun rotate(L, n) =
  if n = length(L) then []
  else
    recursiveRotate([],L,length(L)-n mod length(L));
```

Sample program :

Query :

```
rotate([1, 2, 3, 4, 5], 1);
```

Output :

```
val it = [5,1,2,3,4] : int list
```

4. Merge Sort

```
fun split L =
    let
    val n = length(L) div 2
    in
    (List.take(L,n), List.drop(L,n))
 end;

fun mergeList [ ] L   = L
|   mergeList L [ ] = L
|   mergeList (L1 as H1::T1) (L2 as H2::T2) =
    if H1 < H2
    then
    H1 :: (mergeList T1 L2)
    else
    H2 :: (mergeList L1 T2)

fun msort [E] = [E]
|   msort [ ] = [ ]
|   msort L =
    let
    val (L1,L2) = split L
    in
    mergeList (msort L1) (msort L2)
    end;
```

Sample program :

Query :

```
msort([2, 5, 3, 4, 1]);
```

Output :

```
val it = [1,2,3,4,5] : int list
```

5. Tower of Hanoi

```
fun hanoiSolver(0,_,_,_) = []
|   hanoiSolver(n,PEG3,PEG1,PEG2) =
    hanoiSolver(n-1,PEG2,PEG1,PEG3) @
    [(PEG1,PEG3)] @
    hanoiSolver(n-1,PEG3,PEG2,PEG1)


fun hanoi(n,PEG1,PEG2,PEG3) =
    hanoiSolver(n,PEG3,PEG1,PEG2);
```

Sample program :

Query :

```
hanoi (3,1,2,3);
```

Output :

```
val it = [(1,3),(1,2),(3,2),(1,3),(2,1),(2,3),(1,3)] : (int * int) list
```