Lee Moriarity

Professor Kelly

CIS 112 ONL01

14 September 2022

Lab #3

PART 1: Do EXERCISES #4, #6 & #7 at the end of Chapter 3 on p. 203.

int puzzle(int base, int limit)

{

if (base > limit)

return -1;

else

if (base == limit)

return 1;

else

return base \* puzzle(base + 1, limit);

}

4. Identify

1. The base case(s) of the puzzle method
   1. **if (base > limit)**

**return -1;**

* 1. **if (base == limit)**

**return 1;**

1. The general case(s) of the puzzle method
   1. **if (base < limit)**

**return base \* puzzle(base + 1, limit);**

1. Constraints on the arguments passed to the puzzle method
   1. **base > 0 and limit >= base**

6. Given the following method:

Int exer(int num)

{

if (num == 0)

return 0;

else

return num + exer(num + 1);

}

1. Is there a constraint on the value that can be passed as an argument for this

method to pass the smaller-caller test?

**num <= 1**

1. Is exer(7) a valid call? If so, what is returned from the method?

**No, this will result in an infinite recursion since num will just keep increasing by 1 without ever returning a value.**

1. Is exer(0) a valid call? If so, what is returned from the method?

**Yes, this is a valid call since the condition is true so it will return 0**

1. Is exer(-5) a valid call? If so, what is returned from the method?

**Yes, this is a valid call because it will return -15**

7. For each of the following recursive methods, identify the base case, the general case,

and the constraints on the argument values and explain what the method does.

1. int power(int base, int exponent)

{

if (exponent == 0)

return 1;

return (base \* power(base, exponent-1));

}

**base case: if (exponent == 0)**

**return 0;**

**general case: if (exponent != 0)**

**return (base \* power(base, exponent-1));**

**constraints: exponent = 0**

1. int factorial (int n)

{

if (n > 0)

return (n \* factorial (n - 1));

else

if (n == 0)

return 1;

}

**base case: if (n == 0)**

**return 1;**

**general case: if (n > 0)**

**return (n \* factorial (n - 1));**

**constraints: n < 1?????????????/**

1. int recur(int n)

{

if (n < 0)

return -1;

else if (n < 10)

return 1;

else

return (1 + recur(n / 10));

}

**base case(s): 1. if (n < 0)**

**return -1;**

**2. else if (n < 10)**

**return 1;**

**general case: if (n > 10)**

**return (1 + recur(n / 10));**

**constraints: n > 0 and n > 10**

1. int recur2(int n)

{

if (n < 0)

return -1;

else if (n < 10)

return n;

else

return ((n % 10) + recur2(n / 10));

}

**base case(s): 1. if (n < 0)**

**return -1;**

**2. else if (n < 10)**

**return n;**

**general case: if (n > 0)**

**return (1 + recur(n / 10));**

**constraints: n > 10**