A white background with black text

Description automatically generated

1. The result for calling fact(5) would be 120. Since 5! = 5 \* 4 \* 3 \* 2 \* 1. The function finds this by using the call stack to store 5 \* factorial(4) -> 5 \* 4 \* fact(3) until it gets to fact(1) which would return 1 since it is the identity to multiply the other terms to. Note: fact(1) and fact(0) is the base case that we have since they both will equal 1. This function will do any factorial n>=0 right since if it is negative, a negative factorial is undefined number of permutations.
2. The base case is when n either equals 1 or is less than 1. This is when the recursive calls stop and it returns the multiplicative identity, 1. Since the factorial of any number is n \* n-1 \* … \* 1, it satisfies the base case.
3. The general case is when n > 1. This is because this is where the recursive calls are to make the problem a smaller version of itself until n eventually is 1.

A white background with black text

Description automatically generated

1. The result of fact(-2) would be 1 since n <= 1.