Problem 1 (Recursion) 7 Points

Write a Python function `fastGrow` that accepts as input two integers \( x, y \) and does the following:

- If at least one of \( x \) and \( y \) is not an integer, `fastGrow` raises a `TypeError`.
- If at least one of \( x \) and \( y \) is negative, `fastGrow` raises a `ValueError`.
- In all other cases `fastGrow` returns \( f(x, y) \) where \( f \) is defined as follows:
  - \( f(x, y) = y + 1 \), if \( x = 0 \)
  - \( f(x, y) = f(x - 1, 1) \), if \( y = 0 \)
  - \( f(x, y) = f(x - 1, f(x, y - 1)) \), in all other cases.

*Hints:*
- For testing your program, use small values. The above function grows quite fast.
- The Python function `IsInstance` could be helpful.

Problem 2 (Polynomials) 8 Points

Write a Python class `IntegerPolynomial` as follows:

- Polynomials are generated by providing a list of integer coefficients, with the \( i \)th element in the list being the coefficient of \( x^i \). e.g., to define \( f(x) = 2x^2 + x - 3 \) a user could enter \( f=IntegerPolynomial([-3, 1, 2]) \).
- Provide an implementation of `__str__`, returning a string representation of a polynomial as output by the `prettyprint` function from Exam 1.
- Provide an implementation of `*`, multiplying two polynomials by means of Karatsuba’s algorithm.
- Provide a method `deg()` returning the degree of a polynomial. For the zero polynomial, your function should return Python’s special value `None`.
- Provide an implementation of `==` to check if two polynomials are equal – in particular \( 42 \) and \( 42+0\times x \) must be recognized as equal.

*Good luck—and do not hesitate to ask questions!!*