1.A)

As classes primitivas são as classes que não são definidas a partir de outras classes
Ex:

1. Bounded: Classe que estaberlece limites mínimos e máximos e que instancia os tipos Int, Char, Bool, (), Ordering e tuples. Métodos: minBound :: a ; maxBound :: a
2. Eq: Trata os métodos de igualdade e desigraldade e instancia todos os tipos, com excessão dos tipos IO e funções. Métodos: (==) :: a -> a -> Bool
   (/=) :: a -> a -> Bool
3. Enum: que trata a enumerabilidade, ou seja, define os métodos de operações sobres tipos sequencialmente ordenados instanciando os tipos (), Bool, Char, Int, Integer, Ordering, Float e Double. Métodos: succ, pred :: a -> a; toEnumm :: Int -> a; FromEnum :: a -> Int; enumFrom :: a -> [a]

As classes secundárias são aqueleas definidas a partir de outras classes.
Ex:

1. Ord: Derivado de Eq, define métodos para tipos de dados totalmente ordenados, instanciando todos os tipos, exceto funções () e IOError. Métodos: compara :: a -> a -> Ordering; (<), (<=), (>=), (>) :: a -> a -> Bool; max, min :: a-> a, etc.
2. Num: Derivado de Eq e Show, define os métodos para operações com números e instancias os tipos Int, Integer, Float, e Double. Métodos: negate :: a -> a; abs, signum :: a -> a, enc.
3. Real: Derivado de Ord e Num, define os métodos numéricos de operações. Métodos: toRational :: a -> Rational

1.B)

Real, Integral, Fractional, Floating, RealFrac e RealFloat definem os métodos numéricos de operações. Real instancia os tipos Int, Integer, Float e Double, as outras classes instanciar apenas Float e Double.

2)

Polimorfismo Universal, pode ser aplicado em diversos casos, trabalhando potencialmente num conjunto infinito de tipos de modo disciplinado, possuindo os tipos:

1. Paramétrico: Onde a definição de um elemento por si só é incompleta, assim ela precisa parametrizar o tipo para que ele exista
2. Inclusão: É o polimorfismo básico, quando uma classe Pai aponta para um objeto de classe filho

Polimorfismo Ad-Hoc: é implementado quando queremos definir algo específico, não podendo ser usado em todos lugar:

1. Coerção: é meio para contornar a rigidez dos tipos monomórficos. Existe um mapeamento interno entre os tipos;

2. Sobrecarga: possibilita termos mais de um métodos cmo o mesmo identificador, mas eles devem ter parâmetros diferentes