1. Специальная форма let (обычная, не именнованная) работает следующим образом:
Она сопоставляет имя с соответсвующим ей значением, таким образом, вычисление значениея происходит единственный раз.
(let ((name_1 value_1) (name_2 value_2) ... (name_n value_n)) <тело>)

Примером может послужить следующая ситуация:
    При выполнении программы, необходимо использовать значение выражения несколько раз, например (* 1 2 (sqrt 4)), и для того, чтобы каждый раз не писать и не вычислять это, можно воспользоваться спец. формой let:
    (let ((help_val (* 1 2 (sqrt 4)))) <необходимые действия>)

3. При нормальном порядке пока что-то не понадобится, оно не вычисляется, в аппликативном же вычисляются сначала все аргументы, потом применяется функция.
Преимуществом аппликативного порядка является то, что нет необходимости несколько раз считать одинаковые выражения: например дана функция (define (f a) (* a a)). И необходимо посчитать (f (* 3 4)). в аппликативном порядке (* 3 4) посчитается только один раз, в нормальном порядке сначала подставится (* (* 3 4)(* 3 4)) и посчитается два раза.
Недостатком является то, что для для любой процедуры вычисляются сразу все аргументы, например для логических процедур (or, and) будут считаться сразу все и возможно лишние тоже (при or 1 аргумент будет #t, при and 1 аргумент #f). То есть точно посчитаются все аргументы.