Задача 1

$G : N = \{S, K\}\ T = \{a, b\}\ S = S$

$P : \{S \to aaKb, K \to aK, K \to aKb, K \to \epsilon\}$

Док-во: индукцией по числу шагов покажем что на n-ом шаге грам-ка G порождает элемент из

$$\{a^{n+2}Kb, a^{2+n}Kb^{1+n}, a^{n+1}b, a^{1+n}b^n, a^{2+x+y}Kb^{1+y}, a^{1+x+y}b^y, a^{1+x+y}b^{1+y}$$

, где x+y=n }
1.база: k=1 $S \to aaKb$ верно
2.Пусть для n=k-1 выполняется

$$S \to \{a^{k+1}Kb, a^{k+1}Kb^k, a^kb, a^kb^{k-1}, a^{2+x+y}Kb^{1+y}, a^{1+x+y}b^y, a^{1+x+y}b^{1+y}$$

, где x+y=k-1 }
3. Тогда на k-ом шаге правило может осуществится только для элементов $\{a^{k+1}Kb, a^{k+1}Kb^k, a^{2+x+y}Kb^{1+y}$, где x+y=k-1 } и будут получены элементы:

$$\{a^{k+2}Kb, a^{2+k}Kb^{1+k}, a^{k+1}b, a^{1+k}b^k, a^{2+x+y}Kb^{1+y}, a^{1+x+y}b^y, a^{1+x+y}b^{1+y}$$

, где x+y=k }

Задача 3

Из того, что $n \neq m \Leftrightarrow n < m$ or $m < n \Rightarrow$. Опираемся на з.1
G: N = $\{S_1, S_2, A, B, S\}, T = \{a, b\}, S = S$
P :
$\{\ S \to S_1|S_2, S_1 \to aaAb, A \to aA|aAb|\epsilon, S_2 \to aBbb, B \to Bb|aBb|\epsilon\}$

Доказательство аналогично док-ву из з.1

Задача 4

$a)(aa, bb, ab, ba)^*$
$b)(b)^*(a)^*ab(b)^*(a)^*$
$c)(a|b)^*(a|b)(A|B)(a|b)^*$

Задача 5

РВ = $txt|txtxt|txt(t|x)^*txt$

1. Очевидно, что слова, заданные РВ принадлежат языку, т.к. начинаются и оканчиваются на txt.

2. Всякое слово из языка начинается и оканчивается на txt, и если они не пересекаются, то между ними может содержать любую (в том чимсле пустую)комбинацию из x и t, что равно $(x|t)^*$, т.е. $\forall$ слово из языка может быть задано данным РВ.

Значит, РВ задано корректно.

Задача 6

У нас есть процедура генерации языка. Сохраним все слова языка,если он конечный или мы можем его сохранить. Далее, данное нам слово будем последовательно посимвольно сравнивать со словами нашего языка и при совпадении возвращать "да"или "нет если слов конечное число и совпадения не было. Если же слов в языке не конечное число, то запустим процедуру генерации и каждое новое порожденное слово будем сравнивать со словом, принадлежность которого к языку изучается. В этом случае, кроме результат либо "да либо процедура распознавания не остановится.

Теперь пусть есть алгоритм распознавания. Создадим алгоритм генерации следующим образом: из упорядоченного по длинам слов, а для слов с равными длинами - в лексикографическом порядке - множества $\{a, b\}^*$ будем последовательно подавать слова алгоритму распознавания, затем если он, возвращает результат "да то данное слово возвращаем как сгенерированное.