Zad. 1 Zastosuj metodę rekurencji uniwersalnej do rozwiązania następującego równania rekurencyjnego: $T(n)=3T(\frac{n}{3})+n^2$.

Wzór ogólny to $T(n)=aT(\frac{n}{b})+f(n)$. Stąd $a=3, b=3, f(n)=n^2$.

Najpierw wyliczamy sobie $n^{\log_b a}=n^{\log_3 3}=n$.

Sprawdzamy do którego warunku pasuje nasze wyliczenie, tzn:

1. Jeśli $f(n)=O(n^{1-\varepsilon}), \text{to} T(n)=\Theta(n)$.
2. Jeśli $f(n)=\Theta(n), \text{to} T(n)=\Theta(n\log(n))$.
3. Jeśli $f(n)=\Omega(n^{1+\varepsilon}), \text{to} T(n)=\Theta(f(n))$.

W naszym przypadku zachodzi warunek 3, tzn. $n^2=\Omega(n^{1+\varepsilon})$, zatem $T(n)=\Theta(n^2)$.

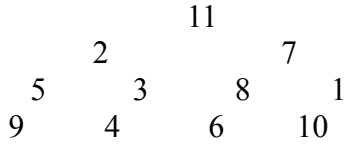
Zad. 3 Jaka jest złożoność QuickSort w najgorszym przypadku, jeśli elementem osiowym jest mediana? Uzasadnij odpowiedź. /2 pkt/

Złożoność dla takiego QuickSorta wynosi w najgorszym przypadku $O(n\log n)$. Jest to związane z tym, że wybierając medianę gwarantujemy podzielenie problemu na dwie równe połowy.

Zad. 4 Podaj algorytm przekształcenia liczby dziesiętnej na binarną bez zapisywania bitów w tablicy (liście itp,). /2 pkt/

```
int func ( int dec )
{
	int bin = 0;
	int d = 1;
	int n = 1;

	while ( dec > pow ( 2, n ) ) n++;	//dzieki temu wiemy ile pozycji zajmie
liczba binarna

	while ( d < pow ( 2, n ) ) d *= 10;	//liczbę binarną zapiszemy w postaci
dziesiętnej
					//d wskazuje największy bit tej liczby ( musimy
iść od				//tylu, tak jak w algorytmie normalnym
byśmy				//zapisywali tablice od początku)

	while ( 1 )
	{
		int tmp = dec%2;
		bin += d * tmp;

		if ( d == 1 ) return bin;
		dec /= 2;
		d /= 10;
	}
}
```

Zad. 5 Wypisz węzły drzewa T=[11, 2, 7, 5, 3, 8, 1, 9, 0, 4, 6, 10, 0, 0] w porządku preorder, inorder i postorder. (0-brak syna) /3 pkt/



preodrer: 11, 2, 5, 9, 3, 4 , 7, 8, 6, 1, 10
inorder: 9, 5, 2, 4, 3, 11, 6, 8, 7, 10, 1
postorder: 9, 5, 4, 3, 2, 6, 8, 10, 1, 7, 11

chyba :D