Task 2 3.2.1.

1

	cost	time	
`function search(value)`			
`{`			
`for array in arrays`	c_1	$\sum_{j=0}^{k-1} b_j$ (or $+1$ if extra cond.)	
`{`			
`int l = 0, r = array.length;`	c_2	$\sum_{j=0}^{k-1} b_j$	
`while (l < r)`	c_3	$\sum_{j=0}^{k-1} b_j \cdot c_j + 1$	c_j $1 \le c_j \le j$ number of shifts of pointers l/r.
`{`			
`int mid = (l + r) / 2;`	c_4	$\sum_{j=0}^{k-1} b_j \cdot c_j$	
`if (array[mid] < value)`	c_5	$\sum_{j=0}^{k-1} b_j \cdot c_j$	
`{ l = mid + 1; }`	c_6	$\sum_{j=0}^{k-1} \sum_{m=1}^{c_j} (t_m)$	t_m - 0/1 if( ) == true
`else if (array[mid] > value)`	c_7	$\sum_{j=0}^{k-1} \sum_{m=1}^{c_j} (1-t_m)$	
`{ r = mid; }`	c_8	$\sum_{j=0}^{k-1} \sum_{\ell=1}^{\sum (1-t_m)} t_\ell$	t_ℓ - 0/1 else if() == true
`else`	c_9	0	
`{ return true; }`	c_{10}	1	
`}`			
`}`			
`return false;`	c_{11}	1	
`}`			

2 Worst-case: binary representation of n - all 1.

$$\sum_{j=0}^{k-1} b_j = k$$

We will not find our value and j times will shift our pointers l and r, so $c_j = j$. Also, every time in while-loop we will execute only c_7 and c_8 ($t_m = 0$ for every m $\sum_{j=0}^{k-1} \sum_{\ell=1}^{\sum(1-t_m)} t_\ell = \sum_{j=0}^{k-1} j$

$$\sum_{j=0}^{k-1} j = \frac{0+k-1}{2} \cdot k = \frac{k^2-k}{2}$$

$$T(n) = c_1 \cdot k + c_2 \cdot k + c_3\left(\frac{k^2-k}{2}+1\right) + c_4\left(\frac{k^2-k}{2}\right) + c_5\left(\frac{k^2-k}{2}\right)$$

$$+ c_7 \cdot \left(\frac{k^2-k}{2}\right) + c_8\left(\frac{k^2-k}{2}\right) + c_{11} = \frac{k^2}{2}\left(c_3 + c_4 + c_5 + c_7 + c_8\right) + k\left(c_1 + c_2\right) -$$

$$- \frac{k}{2}\left(c_3 + c_4 + c_5 + c_7 + c_8\right) + c_3 + c_{11},$$ where $k = \lceil \log(n+1) \rceil$

③ $T(n) = \frac{K^2}{2} \cdot const + K\, const + const = O\left(\log_2^2 (n+1)\right) = O\left(\log_2^2 n\right)$