Алгоритми пошуку даних

1. Існують такі основні алгоритми пошуку даних (кілька відповідей):

	бінарний		з перехрестям		комбінований		послідовний

2. Лінійний пошук необхідного елемента в масиві заснований на:

	упорядкуванні масиву		прямому переборі елементів масиву
	поділенні масиву на дві частини		випадковому виборі елементів масиву

3. Бінарний пошук даних можна використовувати:

	тільки для числових масивів		тільки для масиву рядкового типу
	для упорядкованого масиву		для будь-якого масиву

4. У процесі бінарного пошуку у масиві:

	змінюється тільки ліва межа масиву		межі масиву не змінюються
	змінюються обидві межі		змінюється тільки права межа масиву

5. У процесі реалізації тернарного пошуку масив проміжок поділяється :

	на дві рівні частини		на три рівні частини
	на чотири рівні частини		залежно від умови задачі

6. Розташуйте команди у порядку, що відповідає бінарному пошуку елемента у масиві:

	Програма	Набір команд
1.		`if res<0:` `  print("елемент не знайдено")`
2.		`  a.append(a[i-1]+randint(1,10))` `print(a)`
3.		`while (l<=r):` `  x=(l+r)//2`
4.		`  else:` `  l=x+1`
5.		`  elif b<a[x]:` `  r=x-1`
6.		`else:` `  print(x)`
7.		`b=int(input('Задумай число'))` `l=0; r=n-1; res=-1`
8.		`from random import*` `a=[]; n=10`
9.		`  a.append(randint(1,10))` `  else:`
10.		`for i in range(n):` `  if i==0:`
11.		`  if b==a[x]:` `  res=x; break`

LIVEWORKSHEETS

7. Розташуйте команди у порядку, що відповідає тернарному пошуку максимального значення функції на проміжку:

	Програма	Набір команд
1.		m1 = l + (r − l)/3 m2 = r − (r − l)/3
2.		if (f(m1) < f(m2)): l = m1
3.		x = (l + r)/2 print ('x = ' , x, 'y = ', f(x))
4.		l=-3; r=2; eps=0.0001 while ((r-l)>eps):
5.		else: l=m1; r=m2
6.		elif (f(m1) > f(m2)): r = m2
7.		def f(x): return … # функція

Натисніть FINISH, введіть своє прізвище та ім'я, надішліть вчителю SEND

LIVEWORKSHEETS