Вариант 1

Зад. 1 За целите на тази задача, масив $A[1,\ldots,n]$ се нарича *особен*, ако съществува индекс k, такъв че $1 \le k \le n$ и

$$A[1] < A[2] < \cdots < A[k]$$
$$A[k+1] > A[k+2] > \cdots > A[n]$$

Разгледайте следния алгоритъм. A е особен масив $A[1,\ldots,n]$ от цели числа.

```
MAX(A, ℓ, h)
 1  (* ℓ и h са индекси в A, такива че 1 ≤ ℓ ≤ h ≤ n *)
 2  if ℓ = h
 3      return A[ℓ]
 4  mid ← ⌊(ℓ+h)/2⌋
 5  if A[mid] < A[mid + 1]
 6      return MAX(A, mid + 1, h)
 7  else if A[mid] > A[mid + 1]
 8      return MAX(A, ℓ, mid)
 9  else
10      return A[mid]
```

Докажете, че $\textsc{Max}(A, 1, n)$ връща стойността на максималния елемент на $A[1,\ldots,n]$.

Решение: Такъв масив се нарича *битоничен*, на английски *bitonic*. Ще докажем, че $\textsc{Max}(A, 1, n)$ връща стойността на максималния елемент на $A[1,\ldots,n]$. Тъй като алгоритъмът е рекурсивен, ще докажем това по индукция.

Забележете, че дефиницията не казва нищо за взаимните големини на $A[k]$ и $A[k+1]$. Може $A[k] < A[k+1]$, може $A[k] > A[k+1]$, може $A[k] = A[k+1]$. Следователно, или има точно един, има точно два максимални елемента, които са съседи в масива; като пример, всеки от тези масиви е битоничен:

$$[1, 2, 3, 9, 1]$$
$$[1, 2, 3, 9, 9, 1]$$

Повече от два максимални елемента обаче не може да има. Освен това, ако $A[k] = A[k+1]$, със сигурност това е максималният елемент.

Управляваш параметър за рекурсията е разликата $h - \ell$. Базовият случай е $\ell = h$. В такъв случай алгоритъмът връща $A[\ell]$ на ред 3, което е коректно, понеже единственият елемент е максимален.

Индуктивното предположение е, че за всички стойности на $h-\ell$, такива че $h-\ell < n-1$, алгоритъмът връща коректно максимума на подмасива $A[\ell,\ldots,h]$.

В индуктивната стъпка разглеждаме случая, в който $\ell = 1$ и $h = n$, като $\ell < h$. Изпълнението отива на ред 4, където mid става $\left\lfloor \frac{\ell+h}{2} \right\rfloor$. Преди да продължим с доказателството, да се убедим, че елементите $A[\text{mid}]$ и $A[\text{mid}+1]$ се намират в масива $A[\ell,\ldots,h]$; с други думи, че няма опит за достъп извън подмасива от входа.

- Първо, $\ell \le \text{mid}$, защото $\ell \le \left\lfloor \frac{\ell+h}{2} \right\rfloor$ при $\ell < h$, като равенство е възможно.

- Второ, $\text{mid} < h$, защото

$$\left\lfloor \frac{\ell + h}{2} \right\rfloor < h$$
$$\left\lfloor \frac{\ell + h}{2} - h \right\rfloor < 0$$
$$\left\lfloor \frac{\ell - h}{2} \right\rfloor < 0 \quad // \text{ нека } h = l + k, \text{ където } k > 0$$
$$\left\lfloor \frac{-k}{2} \right\rfloor < 0$$

  като последното е вярно (почти) директно от дефиницията на $\lfloor\ \rfloor$.

Следните възможности са изчерпателни:

- $A[\text{mid}] < A[\text{mid} + 1]$. В такъв случай нито един от $A[\ell], \ldots, A[\text{mid}]$ не е максимален. Тогава максималният елемент се намира в подмасива $A[\text{mid} + 1, \ldots, h]$. Съгласно индуктивното предположение, рекурсивното викане на ред 6 ще върне максимума.
- $A[\text{mid}] > A[\text{mid} + 1]$. В такъв случай нито един от $A[\text{mid} + 1], \ldots, A[h]$ не е максимален. Тогава максималният елемент се намира в подмасива $A[\ell, \ldots, \text{mid}]$. Съгласно индуктивното предположение, рекурсивното викане на ред 8 ще върне максимума.
- $A[\text{mid}] = A[\text{mid} + 1]$. В такъв случай имаме двоен максимум $A[\text{mid}] = A[\text{mid} + 1]$, и алгоритъмът връща точно тази стойност на ред 10.

Зад. 2 Подредете по асимптотично нарастване следните осем функции.

$f_1(n) = n^2 + 40n$ $\quad$ $f_2(n) = n^{\lg n}$ $\quad$ $f_3(n) = 2^n$ $\quad$ $f_4(n) = 2^{\sqrt{n}}$

$f_5(n) = 2^{2+n}$ $\quad$ $f_6(n) = n^{2+\lg n}$ $\quad$ $f_7(n) = \sum_{k=1}^{n!} \frac{k}{2^k}$ $\quad$ $f_8(n) = 11 + \sin(n!)$

Напишете в явен вид самата наредба.

Решение: Ще покажем, че $f_7(n) \asymp 1$. Известно от лекции е, че редът $\sum_{k=1}^{\infty} \frac{k}{2^k}$ е сходящ. Тогава сумата $\sum_{k=1}^{n!} \frac{k}{2^k}$ е ограничена отгоре от константа. Тогава $f_7(n) \asymp 1$.

Очевидно е, че $f_8(n) \asymp 1$, защото синусът взема стойности от интервала $[-1, 1]$, така че цялата сума взема стойности от интервала $[10, 12]$.

Щом $f_7(n) \asymp 1$ и $f_8(n) \asymp 1$, то $f_7(n) \asymp f_8(n)$.

Очевидно $f_1(n)$ е неограничено растяща функция. Тогава $f_8(n) \prec f_1(n)$. Дотук наредбата е

$$f_7(n) \asymp f_8(n) \prec f_1(n)$$

Ще покажем, че $f_1(n) \prec f_2(n)$:

$$\lim_{n\to\infty} \frac{n^{\lg n}}{n^2 + 40n} = \lim_{n\to\infty} \frac{n^{\lg n}}{n^2} = \lim_{n\to\infty} n^{(\lg n)-2} = \infty$$

Дотук наредбата е

$$f_7(n) \asymp f_8(n) \prec f_1(n) \prec f_2(n)$$

Ще покажем, че $f_2(n) \prec f_6(n)$. Наистина, $f_6(n) = n^2 f_2(n)$. Дотук наредбата е

$$f_7(n) \asymp f_8(n) \prec f_1(n) \prec f_2(n) \prec f_6(n)$$

Ще покажем, че $f_6(n) \prec f_4(n)$. Наистина,

$$n^{2+\lg n} \prec 2^{\sqrt{n}} \leftrightarrow \quad // \text{ логаритмуваме}$$
$$(2+\lg n)\lg n \prec \sqrt{n}\lg 2$$

Последното е вярно, понеже всяка полиномиално растяща функция расте по-бързо, в асимптотичния смисъл, от всяка полилогаритмична функция. Дотук наредбата е

$$f_7(n) \asymp f_8(n) \prec f_1(n) \prec f_2(n) \prec f_6(n) \prec f_4(n)$$

Ще покажем, че $f_4(n) \prec f_3(n)$. Наистина,

$$\lim_{n\to\infty} \frac{2^n}{2^{\sqrt{n}}} = \lim_{n\to\infty} 2^{n-\sqrt{n}} = \infty$$

Дотук наредбата е

$$f_7(n) \asymp f_8(n) \prec f_1(n) \prec f_2(n) \prec f_6(n) \prec f_4(n) \prec f_3(n)$$

Ще покажем, че $f_3(n) \asymp f_5(n)$. Наистина, $f_5(n) = 4f_3(n)$. Окончателната наредба тогава е

$$f_7(n) \asymp f_8(n) \prec f_1(n) \prec f_2(n) \prec f_6(n) \prec f_4(n) \prec f_3(n) \asymp f_5(n)$$

Зад. 3 Изследвайте сложността по време като функция на n на следния фрагмент на `C`:

```
int r(int n, int m) {
  int i, s = 0;
  if (n < 2)
    return n*m + 1;
  for(i = 0; i < 3; i ++)
    s += r(n-1,m+i)*r(n-2,m-i);
  s += r(n-1,m) + r(n-2,m-1) + r(n-2, m-2);
  return s; }
```

Решение: Рекурентното уравнение, даващо сложността по време, е

$$T(n) = 4T(n-1) + 5T(n-2) + 1$$

Това е така, понеже управляващата променливата е n. Другата променлива m няма отношение към бързодействието, макар че със сигурност има отношение към върнатия резултат. Забележете, че има $3 + 1 = 4$ викания с първи аргумент $n - 1$ и $3 + 2 = 5$ викания с първи аргумент $n - 2$. Това, че в тялото на `for`-а $r(n-1, m+i)$ се умножава, а не се събира, с $r(n-2, m-i)$, няма значение за броя на извикванията.
Решението е чрез метода на характеристичното уравнение. В случая характеричното уравнение е

$$x^2 - 4x - 5 = 0$$

с корени $x_1 = 5$ и $x_2 = -1$. От нехомогенната част имаме още един корен 1 с кратност 1. Мултимножеството от корените е $\{5, -1, 1\}_M$. Решението е $T(n) \asymp 5^n$.

Вариант 2

Зад. 1 Следният алгоритъм AlgX работи върху сортиран масив $A[1,\dots,n]$ от цели числа.

```
AlgX(A[1,...,n]: сортиран масив от цели числа)
 1  ℓ ← 1, h ← n
 2  while h − ℓ > 1 do
 3      mid ← ⌊(ℓ+h)/2⌋
 4      if A[mid] = mid
 5          return mid
 6      else if A[mid] > mid
 7          h ← mid
 8      else
 9          ℓ ← mid
10  if A[ℓ] = ℓ
11      return ℓ
12  else if A[h] = h
13      return h
14  else
15      return −1
```

Докажете, че AlgX($A[1,\dots,n]$) връща стойност k, такава че $A[k] = k$, ако такава стойност съществува, или -1, ако такава стойност не съществува.

Това е вярно само при вход от два по два различни елементи. Ако има еднакви елементи, алгоритъмът може да бърка. Това е пропуск в условието!

Решение: Ще докажем твърдението с инварианта на цикъла: всеки път, когато изпълнението достигне ред 3, за всеки $k \in \{1,\dots,n\}$, такъв че $A[k] = k$ е вярно, че $k \in \{\ell,\dots,h\}$.

Базата е първото достигане. При него $\ell = 1$ и $h = n$, така че твърдението става "за всеки $k \in \{1,\dots,n\}$, такъв че $A[k] = k$ е вярно, че $k \in \{1,\dots,n\}$", което е тривиално вярно.

Да допуснем, че твърдението е вярно за някое достигане на ред 3, което не е последното. На ред 3, mid приема стойност $\lfloor \frac{\ell+h}{2} \rfloor$. Тъй като $h > \ell$, вярно е, че $\ell \le \lfloor \frac{\ell+h}{2} \rfloor \le h$, така че при адресирането $A[\text{mid}]$ няма опит за адресиране извън $A[\ell,\dots,h]$.

Условието $A[\text{mid}] = \text{mid}$ трябва да е лъжа, в противен случай въпросното достигане на ред 3 щеше да е последното. Следните възможности са изчерпателни:

- $A[\text{mid}] > \text{mid}$. От една страна, това влече, че всеки елемент от $A[\text{mid}+1,\dots,h]$ (също) е по-голям от индекса си.

  Това е така, ако всеки следващ елемент е по-голям от предишния. В противен случай може да не е вярно.

  От това и индуктивното предположение заключаваме, че за всеки $k \in \{1,\dots,n\}$, такъв че $A[k] = k$ е вярно, че $k \in \{\ell,\dots,\text{mid}\}$.

  От друга страна, на ред 7, h получава стойност mid. Тогава при следващото достигане на ред 3 пак е вярно, че всеки $k \in \{1,\dots,n\}$, такъв че $A[k] = k$ е вярно, че $k \in \{\ell,\dots,h\}$.

- $A[\text{mid}] < \text{mid}$. От една страна, това влече, че всеки елемент от $A[\ell,\dots,\text{mid}-1]$ (също) е по-малък от индекса си.

  Аналогичен коментар.

От това и индуктивното предположение заключаваме, че за всеки $k \in \{1, \ldots, n\}$, такъв че $A[k] = k$ е вярно, че $k \in \{\text{mid}, \ldots, h\}$.

От друга страна, на ред 7, ℓ получава стойност mid. Тогава при следващото достигане на ред 3 пак е вярно, че всеки $k \in \{1, \ldots, n\}$, такъв че $A[k] = k$ е вярно, че $k \in \{\ell, \ldots, h\}$.

Да разгледаме терминирането на алгоритъма. Това може да стане по два начина.

1. При някое изпълнение на **while**-а, условието на ред 4 е истина, изпълнява се ред 5 и алгоритъмът терминира, връщайки mid. Това е коректно поведение и дори не ни трябва инвариантата: щом, от една страна, съществува елемент $A[\text{mid}]$, такъв че $A[\text{mid}] = \text{mid}$, а, от друга страна, алгоритъмът връща mid, то неговото поведение в този случай е коректно.

2. При някое достигане на ред 3 е вярно, че $h - \ell \leq 1$. Тогава изпълнението на **while**-а се прекратява и изпълнението отива на ред 10. Има точно две възможности за ℓ и h и ще разгледаме и двете.

   (а) $h - \ell = 1$. В този случай масивът $A[\ell, \ldots, h]$ има два елемента $A[\ell]$ и $A[h] = A[\ell + 1]$. От инвариантата знаем, че ако в $A[1, \ldots, n]$ има елемент, равен на индекса си, той е $A[\ell]$ или $A[h]$. Ако има такъв елемент, една от проверките на редове 10 или 12 ще върне него. Ако няма такъв елемент, нито едно от булевите условия на редове 10 или 12 няма да е истина и алгоритъмът ще върне -1, което е коректно.

   (б) $h - \ell = 0$. В този случай масивът $A[\ell, \ldots, h]$ има един единствен елемент $A[\ell]$. Двете проверки на редове 10 и 12 правят едно и също, но това не е грешно: ако $A[\ell] = \ell$, то още първата проверка ще върне коректно ℓ, в противен случай нито една от двете няма да "сработи" и алгоритъмът ще върне коректно -1 от ред 15.

Зад. 2 Решете следните четири рекурентни уравнения. Допустимо е да ползвате и неформални методи като развиване или дърво на рекурсията.

$$T(n) = 4T(n-1) + 3T(n-2) + 1 \qquad S(n) = 6S\left(\frac{n}{2}\right) + n^2\sqrt{n}$$

$$P(n) = P(\sqrt{n}) + 1 \qquad Q(n) = 2Q\left(\frac{n}{2}\right) + 8Q\left(\frac{n}{4}\right) + n^2$$

Решение: Асимптотиката на $T(n)$ ще намерим чрез метода на характеристичното уравнение. Характеристичното уравнение е

$$x^2 - 4x - 3 = 0$$

с корени $x_1 = 2 + \sqrt{7}$ и $x_2 = 2 - \sqrt{7}$. От нехомогенната част имаме още един корен 1 с кратност 1. Решението е

$$T(n) = A.1^n + B.(2 - \sqrt{7})^n + C.(2 + \sqrt{7})^n$$

за някакви положителни константи A, B и C. Тъй като $|2 - \sqrt{7}| < 1$, вярно е, че

$$\lim_{n \to \infty} (2 - \sqrt{7})^n = 0$$

Следователно, $(2 + \sqrt{7})^n$ определя асимптотиката, така че

$$T(n) = \Theta\left((2 + \sqrt{7})^n\right)$$

Асимптотиката на $S(n)$ ще намерим чрез Мастър теоремата. В терминологията на MT, $a = 6$, $b = 2$ и $f(n) = n^2\sqrt{n} = n^{2.5}$. $\log_b a$ е $\log_2 6$, което е приблизително 2.58. Точната стойност е без значение. Важното е, че $\log_2 6 > 2.5$, което се получава много лесно, имайки предвид факта, че $2^{0.5} = \sqrt{2} < 1.5$:

$$\begin{aligned} \log_2 6 &> 2.5 \ \leftrightarrow \\ 6 &> 2^{2.5} \ \leftrightarrow \\ 4 \times 1.5 &> 4 \times \sqrt{2} \ \leftrightarrow \\ 1.5 &> \sqrt{2} \end{aligned}$$

Щом $f(n) = O(n^{(\log_b a)-\epsilon})$ за някое положително ϵ (да кажем, $\epsilon = 0.01$), първият случай на MT е приложим. Съгласно него, $T(n) \asymp n^{\log_2 6}$.

Асимптотиката на $P(n)$ ще намерим по начин, посочен на лекции. Известно е, че итерирането $n \mapsto \sqrt{n}$ се изпълнява $\Theta(\lg\lg n)$ пъти преди достигане на константа стойност. Това може да се аргументира по много начини.

Един от тях е да заменим n с 2^{2^m}. При това $\sqrt{n}$ става $2^{2^{m-1}}$. Знаем, че итерирането $m \mapsto m-1$ се изпълнява $\Theta(m)$ преди достигане на константна стойност на аргумента. Тъй като $m = \lg\lg n$, веднага следва, че итерирането $n \mapsto \sqrt{n}$ се изпълнява $\Theta(\lg\lg n)$ пъти. И това дава асимптотиката на $P(n)$, а именно $P(n) \asymp \lg\lg n$.

Асимптотиката на $Q(n)$ ще намерим по подобен начин. Заменяме n с 2^m, получавайки

$$Q(2^m) = 2Q\left(2^{m-1}\right) + 8Q\left(2^{m-2}\right) + 4^m$$

Нека $Q(2^m) = Z(m)$. Тогава

$$Z(m) = 2Z(m-1) + 8Z(m-2) + 4^m$$

Това е решимо с метода с характеристичното уравнение. Характеристичното уравнение е

$$x^2 - 2x - 8 = 0$$

с мултимножество от корените $\{-2, 4\}_M$. От нехомогенната част идва още една четворка и мултимножеството от корените става $\{-2, 4, 4\}$. Съгласно изучаваното на лекции, $Z(m) \asymp m4^m$, тоест $Z(m) \asymp m(2^m)^2$. Връщаме се към Q и n: $Q(n) \asymp n^2 \lg n$.

Зад. 3 Изследвайте сложността по време като функция на n на следния фрагмент на `C`:

```
s = 0;
m = n;
while (m >= 1) {
  for (i = 1; i <= floor(n/m); i ++) {
    s ++; }
  m --; }
```

Решение: Броят на изпълненията на тялото на `for`-а задава асимптотиката.

Променливата `m` приема последователно стойностите n, $n-1$, $n-2$, $\ldots 2$, 1, и при последното достигане на `while`-а има стойност 0. Тялото на `for`-а се изпълнява `floor(n/m)` пъти за всяка стойност на `m`.

Тогава сумата

$$S(n) = \left\lfloor \frac{n}{n} \right\rfloor + \left\lfloor \frac{n}{n-1} \right\rfloor + \left\lfloor \frac{n}{n-2} \right\rfloor + \cdots + \left\lfloor \frac{n}{3} \right\rfloor + \left\lfloor \frac{n}{2} \right\rfloor + \left\lfloor \frac{n}{1} \right\rfloor$$

дава асимптотиката на сложността по време на фрагмента.

Асимптотиката на $S(n)$ не се променя, ако просто изпуснем всички "$\lfloor\ \rfloor$":

$$S(n) \asymp \frac{n}{n} + \frac{n}{n-1} + \frac{n}{n-2} + \cdots + \frac{n}{3} + \frac{n}{2} + \frac{n}{1} = n\left(\frac{1}{n} + \frac{1}{n-1} + \cdots + \frac{1}{2} + \frac{1}{1}\right)$$

Съгласно изучаваното на лекции,

$$\frac{1}{n} + \frac{1}{n-1} + \cdots + \frac{1}{2} + \frac{1}{1} \asymp \lg n$$

откъдето $S(n) \asymp n \lg n$.