АНАЛИЗ НА АЛГОРИТМИ — КОРЕКТНОСТ И СЛОЖНОСТ

КОНТРОЛНО № 1 ПО “ДИЗАЙН И АНАЛИЗ НА АЛГОРИТМИ” — СУ, ФМИ
(ЗА СПЕЦИАЛНОСТ “КОМПЮТЪРНИ НАУКИ”, 1. ПОТОК; 14 МАРТ 2018 Г.)

ВАРИАНТ № 1

Разглеждаме два алгоритъма за разпознаване на прости числа — `isPrime1` и `isPrime2`. Входът и на двата алгоритъма е цяло положително число $n \geq 2$.

```
isPrime1(n)  // n ≥ 2
for k ← 2 to n - 1 do
  if n се дели на k
    return false
return true
```

```
isPrime2(n)  // n ≥ 2
return not hasDivisor(n, 2, n-1)

hasDivisor(n, k, r)
// проверява дали n има делител
// между k и r включително (k ≤ r)
if k = r
  if n се дели на k
    return true
  else
    return false
if k + 7 > r
  for j = k to r do
    if n се дели на j
      return true
  return false
for j = 1 to 6 do
  if hasDivisor(n, k+(j-1)⌊(r-k)/7⌋, k+j⌊(r-k)/7⌋)
    return true
return hasDivisor(n, k+6⌊(r-k)/7⌋, r)
```

Задача 1. Докажете, че `isPrime1` е коректен алгоритъм, като формулирате и докажете подходяща инварианта на цикъла. **(15 точки)**

Задача 2. Докажете по индукция, че `hasDivisor` е коректен алгоритъм. **(15 точки)**
(От това следва, че и `isPrime2` е коректен алгоритъм.)
Отбележете изрично типа и параметъра на индукцията.

Задача 3. Намерете сложността по време на алгоритъма `isPrime2`. **(10 точки)**

Задача 4. Задачата за търговския пътник може да се реши с различни алгоритми:
— Пълно изчерпване. Неговата времева сложност е $\Theta(n!)$, където n е броят на градовете.
— Алгоритъм на Хелд—Карп. Времевата сложност на този алгоритъм е $\Theta(n^2.2^n)$.
Кой от двата алгоритъма е по-бърз? Отговорът да се обоснове подробно! **(10 точки)**

АНАЛИЗ НА АЛГОРИТМИ — КОРЕКТНОСТ И СЛОЖНОСТ

КОНТРОЛНО № 1 ПО "ДИЗАЙН И АНАЛИЗ НА АЛГОРИТМИ" — СУ, ФМИ
(ЗА СПЕЦИАЛНОСТ "КОМПЮТЪРНИ НАУКИ", 1. ПОТОК; 14 МАРТ 2018 Г.)

ВАРИАНТ № 2

Разглеждаме два алгоритъма за събиране на редица от n цели числа, $n \geq 1$

```
ALG-1(A[1...n] : integers)
i ← 1, j ← 2, s ← 0
while i < n do
    s ← s + A[i] − 2A[j]
    i ← i + 1
    j ← j + 1
s ← s − 2A[1] + A[n]
return −s
```

```
ALG-2(A[1...n]) // n ≥ 1
return sum(A[1...n])

sum(A[k...r])
// пресмята сбора на елементите с индекси
// между k и r включително (k ≤ r)
if k = r
  return A[k]
if k + 3 > r
  s ← 0
  for j = k to r do
    s ← s + A[j]
  return s
s1 ← sum(A[k...k+⌊(r−k)/3⌋])
s2 ← sum(A[k+⌊(r−k)/3⌋+1...k+2⌊(r−k)/3⌋])
s3 ← sum(A[k+2⌊(r−k)/3⌋+1...r])
return s1 + s2 + s3
```

Задача 1. Докажете, че ALG-1 е коректен алгоритъм,
като формулирате и докажете подходяща инварианта на цикъла. **(15 точки)**

Задача 2. Докажете по индукция, че sum е коректен алгоритъм. **(15 точки)**
(От това следва, че и ALG-2 е коректен алгоритъм.)
Отбележете изрично типа и параметъра на индукцията.

Задача 3. Намерете сложността по време на алгоритъма ALG-2. **(10 точки)**

Задача 4. Детерминантата на числова матрица с n реда и n стълба може да се пресмята по различни начини:

— По формулата от определението за детерминанта. Това изисква събиране на произведения по всички пермутации на стълбове, затова сложността му е $\Theta\big((n+1)!\big)$.

— Чрез привеждане в триъгълен вид. Времевата сложност на този метод е $\Theta\left(n^3\right)$.

Кой от двата начина е по-бърз? Отговорът да се обоснове подробно! **(10 точки)**

А Н А Л И З   Н А   А Л Г О Р И Т М И — К О Р Е К Т Н О С Т   И   С Л О Ж Н О С Т

КОНТРОЛНО № 1 ПО "ДИЗАЙН И АНАЛИЗ НА АЛГОРИТМИ" — СУ, ФМИ

(ЗА СПЕЦИАЛНОСТ "КОМПЮТЪРНИ НАУКИ", 1. ПОТОК; 14 МАРТ 2018 Г.)

В А Р И А Н Т № 3

```
НОД(a,b)
while a ≠ b do
  while a > b do
    a ← a - b
  while b > a do
    b ← b - a
return a
```

```
sumMatrices(A,B,C: matrices n × n)  // n ≥ 1
return sumM(A,B,C,1,n,1,n)  // C ← A + B

sumM(A,B,C,k,r,p,q)  // k ≤ r, p ≤ q
// C[i,j] ← A[i,j] + B[i,j],
// където i се изменя от k до r включително,
// а j се изменя от p до q включително
if (k > r) or (p > q)
  return  // do nothing: empty submatrix
else if (k = r) and (p = q)
  C[k,p] ← A[k,p] + B[k,p]
else
```

$$\mathrm{sumM}\left(A, B, C, k, \left\lfloor\frac{k+r}{2}\right\rfloor, p, \left\lfloor\frac{p+q}{2}\right\rfloor\right)$$

$$\mathrm{sumM}\left(A, B, C, k, \left\lfloor\frac{k+r}{2}\right\rfloor, \left\lfloor\frac{p+q}{2}\right\rfloor+1, q\right)$$

$$\mathrm{sumM}\left(A, B, C, \left\lfloor\frac{k+r}{2}\right\rfloor+1, r, p, \left\lfloor\frac{p+q}{2}\right\rfloor\right)$$

$$\mathrm{sumM}\left(A, B, C, \left\lfloor\frac{k+r}{2}\right\rfloor+1, r, \left\lfloor\frac{p+q}{2}\right\rfloor+1, q\right)$$

Задача 1. Докажете с подходяща инварианта, че алгоритъмът НОД пресмята най-голямия общ делител на целите положителни числа a и b. **(15 точки)**

Задача 2. Докажете по индукция, че sumM записва в матрицата C[k...r, p...q] сбора на числовите матрици A[k...r, p...q] и B[k...r, p...q].
(От това следва, че sumMatrices пресмята сбора на квадратни числови матрици с n реда и n стълба.)
Отбележете изрично типа и параметъра на индукцията. **(15 точки)**

Задача 3. Намерете сложността по време на алгоритъма sumMatrices. **(10 точки)**

Задача 4. Два алгоритъма имат времеви сложности $\Theta\left(n^{n}\right)$ и $\Theta\left(5^{n^{3}}\right)$.

Кой от двата алгоритъма е по-бърз? Отговорът да се обоснове подробно! **(10 точки)**

А Н А Л И З   Н А   А Л Г О Р И Т М И — К О Р Е К Т Н О С Т   И   С Л О Ж Н О С Т

КОНТРОЛНО № 1 ПО "ДИЗАЙН И АНАЛИЗ НА АЛГОРИТМИ" — СУ, ФМИ
(ЗА СПЕЦИАЛНОСТ "КОМПЮТЪРНИ НАУКИ", 1. ПОТОК ; 14 МАРТ 2018 Г.)

В А Р И А Н Т № 4

Разглеждаме два алгоритъма, които пресмятат произведението на целите числа от a до b включително, където a ≤ b са две цели числа.

```
product1(a,b)
p ← 1
for k ← a to b do
  p ← p × k
return p
```

```
product2(a,b)
if a = b
  return a
else
  q ← product2(a, ⌊(a+b)/2⌋)
  r ← product2(⌊(a+b)/2⌋+1, b)
  return q × r
```

Задача 1. С подходяща инварианта докажете коректността на product1. **(15 точки)**

Задача 2. Чрез индукция докажете коректността на product2.
Отбележете изрично типа и параметъра на индукцията. **(15 точки)**

Задача 3. Намерете сложността по време на алгоритъма product2. **(10 точки)**

Задача 4. Два алгоритъма имат времеви сложности $\Theta\left(\left(n^{n}\right)!\right)$ и $\Theta\left(\left(n!\right)^{n}\right)$.

Кой от двата алгоритъма е по-бърз? Отговорът да се обоснове подробно! **(10 точки)**

К Р А Т К И   Р Е Ш Е Н И Я

В А Р И А Н Т   № 1

Задача 1. Инварианта на цикъла на `isPrime1`:

`n` не се дели на никое от числата $1,\ 2,\ 3,\ \ldots,\ k-1$.

Задача 2 се решава със силна индукция по `r – k`.

Задача 3. Времевата сложност $T(n)$ на алгоритъма `isPrime2` удовлетворява уравнението $T(n) = 7T\left(\dfrac{n}{7}\right) + 1$, чието решение се получава от мастър-теоремата: $T(n) = \Theta\big(n\big)$.

Задача 4 се решава чрез логаритмуване:

$$\log(n!) = \Theta(n \log n); \qquad \log(n^2.2^n) = 2 \log n + n \log 2 = \Theta(n).$$

С помощта на граници се доказва, че

$$n = \mathrm{o}(n \log n), \ \text{ тоест } \ \log(n^2.2^n) = \mathrm{o}(\log(n!)).$$

Понеже $n \log n \to \infty$ при $n \to \infty$, то следва, че

$$n^2.2^n = \mathrm{o}(n!).$$

Извод: Алгоритъмът на Хелд—Карп е по-бърз от пълното изчерпване.

В А Р И А Н Т   № 2

Задача 1. Инварианта на цикъла на `ALG-1`:

`s = A[1] – ( A[2] + A[3] + ... + A[i] ) – A[i]` и `j = i + 1`.

Задача 2 се решава със силна индукция по `r – k`.

Задача 3. Времевата сложност $T(n)$ на алгоритъма `ALG-2` удовлетворява уравнението $T(n) = 3T\left(\dfrac{n}{3}\right) + 1$, чието решение се получава от мастър-теоремата: $T(n) = \Theta\big(n\big)$.

Задача 4 се решава чрез логаритмуване:

$$\log\big((n+1)!\big) = \Theta\big((n+1) \log(\mathrm{n}+1)\big) = \Theta(n \log n); \qquad \log(n^3) = 3 \log n = \Theta(n).$$

С помощта на граници се доказва, че

$$n = \mathrm{o}(n \log n), \ \text{ тоест } \ \log(n^3) = \mathrm{o}\Big(\log\big((n+1)!\big)\Big).$$

Понеже $n \log n \to \infty$ при $n \to \infty$, то следва, че

$$n^3 = \mathrm{o}\big((n+1)!\big).$$

Извод: Привеждането в триъгълен вид е по-бързо от определението.

В А Р И А Н Т № 3

Задача 1. Инварианта на цикъла на `НОД`**:**
Най-големият общ делител на `a` и `b` е равен на най-големия общ делител на `A` и `B`, където `A` и `B` са началните стойности на `a` и `b` съответно.

Задача 2 се решава със силна индукция по `(r – k) + (q – p)`.

Задача 3. Времевата сложност $T(n)$ на `sumMatrices` удовлетворява уравнението $T(n) = 4T\left(\frac{n}{2}\right) + 1$, чието решение се получава от мастър-теоремата: $T(n) = \Theta\left(n^2\right)$.

Задача 4 се решава чрез логаритмуване:

$$\log\left(n^n\right) = n \log n; \qquad \log\left(5^{n^3}\right) = n^3 . \log 5 = \Theta(n^3).$$

С помощта на граници се доказва, че

$$n \log n = \mathrm{o}(n^3), \text{ тоест } \log(n^n) = \mathrm{o}\left(\log\left(5^{n^3}\right)\right).$$

Понеже $n^3 \to \infty$ при $n \to \infty$, то следва, че

$$n^n = \mathrm{o}\left(5^{n^3}\right).$$

Извод: По-бърз е алгоритъмът със сложност $\Theta\left(n^n\right)$.

В А Р И А Н Т № 4

Задача 1. Инварианта на цикъла на `product1`**:**

`p = a(a+1)(a+2)(a+3)...(k-1).`

Задача 2 се решава със силна индукция по `b – a`.

Задача 3. Времевата сложност $T(n)$ на `product2` удовлетворява уравнението $T(n) = 2T\left(\frac{n}{2}\right) + 1$, където $n = b - a$. От мастър-теоремата следва, че $T(n) = \Theta\left(n\right)$.

Задача 4 се решава чрез логаритмуване:

$$\log\left(\left(n^n\right)!\right) = \Theta\left(n^n \log\left(n^n\right)\right) = \Theta\left(n^{n+1} \log n\right); \quad \log\left((n!)^n\right) = n \log(n!) = \Theta\left(n^2 \log n\right).$$

С помощта на граници се доказва, че $n^2 = \mathrm{o}(n^{n+1})$.

Умножаваме по $\log n$: $n^2 \log n = \mathrm{o}(n^{n+1} \log n)$, тоест $\log\left((n!)^n\right) = \mathrm{o}\left(\log\left(\left(n^n\right)!\right)\right)$.

Понеже $(n^n)! \to \infty$ при $n \to \infty$, то следва, че $(n!)^n = \mathrm{o}\left(\left(n^n\right)!\right)$.

Извод: По-бърз е алгоритъмът със сложност $\Theta\left((n!)^n\right)$.