

1. Máme dva obrázky - pokaždé jde o čtverec se 100 body uvnitř. V jednom případě byly souřadnice každého bodu generovány nezávisle a pseudonáhodně, ve druhém případě byly generovány analogicky, ale navíc byly souřadnice systematicky modifikovány (nám neznámým) způsobem. Odhadněte, kterého obrázku se týkají dodatečné úpravy a zdůvodněte svůj odhad.

2. Máte jednu hrací kostku. Popište, jak využijte házení kostkou tak, abyste měli generátor náhodných celých čísel v rozmezí 0..10. Všechna čísla 0, 1, 2, …, 10 musí být generována se stejnou pravděpodobností.

3. Vysvětlete, jak pomocí generátoru náhodných čísel zamícháte do náhodného pořadí seřazené pole čísel. Akce musí proběhnout v čase úměrném délce pole.

4. Ověřte, zda lineární kongruenční generátor s danými parametry má maximální možnou délku periody.

A) *xn*+1 = (91 *xn* + 49) mod 600 C) *xn*+1 = (37 *xn* + 55) mod 144

B) *xn*+1 = (8 *xn* + 80) mod 49 D) *xn*+1 = (99 *xn* + 81) mod 113

5. Určete délku periody v Lehmerově generátoru, který je dán předpisem *xn*+1 = ((M−1)∙*xn*) mod M, kde M je prvočíslo.

6. Určete, kolik přibližně prvočísel leží v intervalu

A) [0, 109], B) [109, 2∙109], C) [2∙109, 3∙109]?

7. Řekneme, že přirozené číslo je poloprvočíslo, pokud je buď prvočíslem nebo celou mocninou prvočísla.

Popište modifikaci Eratosthenova síta, která bude generovat právě poloprvočísla. Napište pseudokód.

8. Jako skoroprvočísla označíme právě ta přirozená čísla, která jsou součinem dvou různých prvočísel.

Popište modifikaci Eratosthenova síta, která bude generovat skoroprvočísla. Napište pseudokód.

9. Vygenerujeme všech 168 prvočísel menších než 1000 (2, 3, 5, …, 991, 997) a dále ze seznamu S = (1000, 1001, …, 999999) všech celých čísel větších než 999 a menších než 1000000 vyškrtneme všechny násobky uvedených prvočísel. Kolik čísel zbyde v S a kolik z nich budou prvočísla? Uveďte co nepřesnější odhad.

10. Určete, jaký je maximální možný počet prvočísel v kterémkoli z intervalů [30*k*, 30*k*+29], *k* = 1, 2, 3, 4, ... .

11. Vypočtěte A) GCD(220, 284), B) GCD , C) GCD(2100, 100!)

12. Modulární umocňování, příklad.

1889 mod 11 =

= 181011001B mod 11 =

= 181000000B + 10000B + 1000B + 1B  mod 11 =

= (181000000B ∙ 1810000B ∙181000B ∙181B ) mod 11 =

= ((181000000B mod 11) ∙ (1810000B mod 11) ∙ (181000B mod 11) ∙ (181B mod 11)) mod 11. (\*\*)

Mezivýpočet:

181B mod 11 = **7**

1810B mod 11 = ((181B mod 11) ∙ (181B mod 11)) mod 11 = (7∙7) mod 11 = 5

18100B mod 11 = ((1810B mod 11) ∙ (1810B mod 11)) mod 11 = (5∙5) mod 11 = 3

181000B mod 11 = ((18100B mod 11) ∙ (18100B mod 11)) mod 11 = (3∙3) mod 11 = 9

1810000B mod 11 = ((181000B mod 11) ∙ (181000B mod 11)) mod 11 = (9∙9) mod 11 = 4

18100000B mod 11 = ((1810000B mod 11) ∙ (1810000B mod 11)) mod 11 = (4∙4) mod 11 = 5

181000000B mod 11 = ((18100000B mod 11) ∙ (18100000B mod 11)) mod 11 = (5∙5) mod 11 = 3

Návrat do (\*\*):

((181000000B mod 11) ∙ (1810000B mod 11) ∙ (181000B mod 11) ∙ (181B mod 11)) mod 11 = (3∙4∙9∙7) mod 11 = 8.

Závěr

1889 mod 11 = 8.

13. Analogickým postupem jako v příkladu modulárního umocňování vypočtěte:

A) 18189 mod 11 B) 2100 mod 20 C) 850 mod 7 D) 123456 mod 1000

14. Uvedený kód počítá celočíselnou mocninu *xn*. Popište, jak jej upravíte, aby počítal *xn* mod *m*, pro kladné celé *m*.

Minimalizujte riziko přetečení

BinPower(int *x*, int *n*) {

int *r* = 1, *y* = *x;*

while (*n* > 1) {

if (n % 2 == 1) *r* \*= *y;*

*y* \*= *y;*

*n* /= 2;

}

return *r\*y;*

}