Lucrare de laborator nr. 6

FUNCȚII HASH ȘI SEMNĂTURI DIGITALE

Sarcina 1. Studiați materiale didactice recomandate la temă plasate pe ELSE.

Sarcina 2. **Utilizând platforma *wolframalpha.com* sau aplicația *Wolfram Mathematica*, generați cheile, realizați semnarea și validarea semnăturii digitale a mesajului *m* pe care l-ați obținut realizând lucrarea de laborator nr. 2.**
Semnarea va fi realizată aplicând semnătura RSA. Valoarea lui *n* trebuie să fie de cel puțin 3072 biți. Algoritmul hash va fi selectat din lista de mai jos, în conformitate cu formula *i* = (*k* mod 24) +1, unde *k* este numărul de ordine al studentului în lista grupei, *i* este indicele funcției hash din listă:

1. MD4	7. SHA-1	13. SHA3-256	19. RipeMD-320
2. MD5	8. SHA-224	14. SHA3-384	20. Whirlpool
3. MD2	9. SHA-256	15. SHA3-512	21. NTLM
4. MD6-128	10. SHA-384	16. RipeMD-128	22. Haval192,3
5. MD6-256	11. SHA-512	17. RipeMD-160	23. Haval224,4
6. MD6-512	12. SHA3-224	18. RipeMD-256	24. Haval256,4

Sarcina 3. **Utilizând platforma *wolframalpha.com* sau aplicația *Wolfram Mathematica*, realizați semnarea și validarea semnăturii digitale a mesajului *m* pe care l-ați obținut realizând lucrarea de laborator nr. 2.**
Semnarea va fi realizată aplicând semnătura ElGamal (*p* și *generatorul* sunt dați mai jos). Algoritmul hash va fi selectat din lista de mai jos, în conformitate cu formula *i* = (*k* mod 24) +1, unde *k* este numărul de ordine al studentului în lista grupei, *i* este indicele funcției hash din listă:

1. NTLM	7. MD6-512	13. SHA3-224	19. RipeMD-256
2. MD4	8. SHA-1	14. SHA3-256	20. RipeMD-320
3. MD5	9. SHA-224	15. SHA3-384	21. Whirlpool
4. MD2	10. SHA-256	16. SHA3-512	22. Haval192,3
5. MD6-128	11. SHA-384	17. RipeMD-128	23. Haval224,4
6. MD6-256	12. SHA-512	18. RipeMD-160	24. Haval256,4

Notă:

- **Pentru sarcinile 2 și 3 utilizați reprezentarea numerică zecimală a valorii hash. Pentru comoditate, la conversia dintr-o bază în alta, puteți să folosiți pagina https://www.rapidtables.com/convert/number/hex-to-decimal.html.**

- **Pentru sarcina 2 considerați:**
  p=32317006071311007300153513477825163362488057133489075174588434139269806834136210002792056362640164685458556357935330816928829023080573472625273554742461245741026202527916572972862706300325263428213145766931414223654220941111348629991657478268034230553086349050635557712219187890332729569696129743856241741236237225197346402691855797767976823014625397933058015226858730761197532436467475855460715043896844940366130497697812854295958659597567051283852132784468522925504568272879113720098931873959143374175837826000278034973198552060607533234122603254684088120031105907484281003994966956119696956248629032338072839127039, **care are 2048 biți și generatorul** g=2.

- **Raportul trebuie însoțit de comentarii detaliate în realizarea tuturor pașilor algoritmilor.**

Funcții utile în Wolfram:

- ***Prime*[n]** – *returnează al n-lea număr prim din lista numerelor prime (n este mărginit);*
- ***RandomPrime*[{i_{min}, i_{max}}]** – *returnează un număr prim pseudoaleator cuprins între* i_{min} *și* i_{max} *;*
- ***RandomInteger*[i_{max}]** – *returnează un număr întreg pseudoaleator cuprins între 0 și* i_{max} *;*
- ***Mod[a, n]*** – *returnează restul împărțirii lui a la n;*
- ***PoerMod[a, b, n]*** – *returnează restul împărțirii lui* a^b *la n;*
- ***FactorInteger [n]*** – *returnează lista de factori primi ai lui n, împreună cu exponenții lor;*
- ***IntegerDigits[n, b]*** – *returnează lista de cifre în baza b a numărului întreg n;*
- ***Length[lst]*** – *returnează lungimea listei lst;*