Bevezetés a matematikába

Járai Antal

Ezek a programok csak szemléltetésre szolgálnak.

▶ 1. Halmazok

▶ 2. Természetes számok

▶ 3. A számfogalom bővítése

▶ 4. Véges halmazok

▶ 5. Végtelen halmazok

▶ 6. Számelmélet

▶ 7. Gráfelmélet

▶ 8. Algebra

▼ 9. Kódolás

▼ 9.1. Kommunikáció és kódolás


```
> restart;
```

9.1.1. Információ, bit, entrópia.

```
> H:=proc(F::list(positive),r::positive) local i,s,h; s:=0;
  h:=0;
  for i to nops(F) do s:=s+F[i]; od;
  for i to nops(F) do h:=h-log[r](F[i]/s)*F[i]/s; od;
  h; end;

  H([1,1],2); H([1,3],2); evalf(%);
```

$$H := \mathbf{proc}(F::(list(positive)), r::positive)$$

```
    local i, s, h;
    s := 0;
    h := 0;
    for i to nops(F) do
        s := s + F[i]
    end do;
    for i to nops(F) do
        h := h - log[r](F[i] / s) * F[i] / s
    end do;
    h
end proc
```

$$\frac{1}{2} - \frac{3}{4}\,\frac{\ln\left(\frac{3}{4}\right)}{\ln(2)}$$

$$0.8112781245 \qquad (9.1.1.1)$$

```
> with(StringTools);
```

[*Anagrams*, *AndMap*, *ApproximateSearch*, *ApproximateSearchAll*, *ArithmeticMean*, *Border*, *BorderArray*, *BorderLength*, *CamelCase*, *Capitalize*, *CaseJoin*, *CaseSplit*, *Center*, *Centre*, *Char*, *CharacterFrequencies*, *CharacterMap*, *Chomp*, *Chop*, *CommonPrefix*, *CommonSuffix*, *Compare*, *CompareCI*, *CountCharacterOccurrences*, *Decode*, *Delete*, *Drop*, *EditDistance*, *Encode*, *Entropy*, *Escape*, *Exchange*, *ExpandCharacterClass*, *ExpandTabs*, *Explode*, *Fence*, *Fibonacci*, *Fill*, *FirstFromLeft*, *FirstFromRight*, *FormatMessage*, *FormatTime*, *FromByteArray*, *Generate*, *Group*, *HammingDistance*, *HammingSearch*, *HammingSearchAll*, *HasASCII*, *HasAlpha*, *HasAlphaNumeric*, *HasBinaryDigit*, *HasControlCharacter*, *HasDigit*, *HasGraphic*, *HasHexDigit*, *HasIdentifier*, *HasIdentifier1*, *HasLower*, *HasOctalDigit*, *HasPrintable*, *HasPunctuation*, *HasSpace*, *HasUpper*, *HasVowel*, *Hash*, *Implode*, *Insert*, I, *IsASCII*, *IsAlpha*, *IsAlphaNumeric*, *IsAnagram*, *IsBalanced*, *IsBinaryDigit*, *IsConjugate*, *IsControlCharacter*, *IsDigit*, *IsGraphic*, *IsHexDigit*, *IsIdentifier*, *IsIdentifier1*, *IsLower*, *IsMonotonic*, *IsOctalDigit*, *IsPalindrome*, *IsPeriod*, *IsPermutation*, *IsPrefix*, *IsPrimitive*, *IsPrintable*, *IsPunctuation*, *IsSorted*, *IsSpace*, *IsSubSequence*, *IsSuffix*, *IsUpper*, *IsVowel*, *Join*, *LeftFold*, *LeftRecursivePathOrder*, (9.1.1.2)

Length, *Levenshtein*, *LexOrder*, *LongestCommonSubSequence*, *LongestCommonSubString*, *LowerCase*, *LyndonFactors*, *Map*, *MaxChar*, *MaximalPalindromicSubstring*, *Metaphone*, *MinChar*, *MinimumConjugate*, *MonotonicFactors*, *NGrams*, *NthWord*, *OrMap*, *Ord*, *Overlap*, *PadLeft*, *PadRight*, *ParseTime*, *PatternDictionary*, *Period*, *Permute*, *PrefixDistance*, *PrimitiveRoot*, *Random*, *Randomize*, *RegMatch*, *RegSplit*, *RegSub*, *RegSubs*, *Remove*, *Repeat*, *RevLexOrder*, *Reverse*, *RightFold*, *RightRecursivePathOrder*, *Rotate*, *Search*, *SearchAll*, *Select*, *SelectRemove*, *Shift*, *ShortLexOrder*, *ShortRevLexOrder*, *SimilarityCoefficient*, *Sort*, *Soundex*, *Split*, *Squeeze*, *Stem*, *StringBuffer*, *SubString*, *Substitute*, *SubstituteAll*, *SuffixDistance*, *Support*, *SyllableLength*, *Tabulate*, *Take*, *ThueMorse*, *ToByteArray*, *Trim*, *TrimLeft*, *TrimRight*, *UpperCase*, *Visible*, *WildcardMatch*, *WordCount*, *Words*, *WrapText*]

```
> Entropy("ab"); Entropy("aaab");
```

$$1.$$

$$0.811278124459132832 \qquad (9.1.1.3)$$

▶ **9.1.2. Segédtétel.*

▶ *9.1.3. Tétel.*

▼ *->9.1.4. Feladat.*

▼ *->9.1.5. Feladat.*

▼ *->9.1.6. Feladat.*

▶ *9.1.7. Kódolás.*

▼ 9.2. Gazdaságos kódolás

▶ *9.2.1. Betűnkénti kódolás.*

▼ *9.2.2. Példa.*

```
> M:=table():
  M["a"]:="10111": M["b"]:="111010101": M["c"]:=
  "11101011101":
  M["d"]:="1110101": M["e"]:="1": M["f"]:="101011101":
  M["g"]:="111011101": M["h"]:="1010101": M["i"]:="101":
  M["j"]:="1011101110111": M["k"]:="111010111": M["l"]:=
  "101110101":
```

```
M["m"]:="1110111": M["n"]:="11101": M["o"]:="11101110111":
M["p"]:="10111011101": M["q"]:="1110111010111": M["r"]:=
"1011101":
M["s"]:="10101": M["t"]:="111": M["u"]:="1010111":
M["v"]:="101010111": M["w"]:="101110111": M["x"]:=
"11101010111":
M["y"]:="1110101110111": M["z"]:="11101110101": M[" "]:="":
M["0"]:="1110111011101110111": M["1"]:="101110111101110111":
M["2"]:="101011101110111": M["3"]:="1010101110111":
M["4"]:="10101010111": M["5"]:="101010101": M["6"]:=
"11101010101":
M["7"]:="1110111010101": M["8"]:="111011101110101":
M["9"]:="11101110111011101": M["."]:="10111010111010111":
M[","]:="11101110101011101110111": M["?"]:="101011101110101":
M[":"]:="11101110111010101": M["-"]:="111010101010111":
```

```
> ASCII2Morse:=proc(s::string) local i,m; m:="";
  for i to length(s) do m:=cat(m,M[s[i]],"000") od; m; end;

  ASCII2Morse("ad "); ASCII2Morse("emi ");
```

$$ASCII2Morse := \mathbf{proc}(s::string)$$
$$\quad \mathbf{local}\ i, m;$$
$$\quad m := "";$$
$$\quad \mathbf{for}\ i\ \mathbf{to}\ length(s)\ \mathbf{do}$$
$$\quad\quad m := cat(m, M[s[i]], "000")$$
$$\quad \mathbf{end\ do};$$
$$\quad m$$
$$\mathbf{end\ proc}$$

"10111000111010100000"

"1000111011100010100000" (9.2.2.1)

```
> addcomma:=proc(A,M,Ct,p) local i;
  for i to length(A) do Ct[A[i]]:=cat(M[A[i]],p) od; end;
```

$$addcomma := \mathbf{proc}(A, M, Ct, p)$$ (9.2.2.2)
$$\quad \mathbf{local}\ i;$$
$$\quad \mathbf{for}\ i\ \mathbf{to}\ length(A)\ \mathbf{do}$$
$$\quad\quad Ct[A[i]] := cat(M[A[i]], p)$$
$$\quad \mathbf{end\ do}$$
$$\mathbf{end\ proc}$$

```
> Ct:=table(): A:="abcdefghijklmnopqrstuvwxyz 0123456789.,?:-
  ";
  addcomma(A,M,Ct,"000"):
```

$$A := "abcdefghijklmnopqrstuvwxyz\ 0123456789.,?:-"$$ (9.2.2.3)

```
> Ct["m"];
```

"1110111000" (9.2.2.4)

```
> code:=proc(m,Ct) local i,c; c:="";
  for i to length(m) do c:=cat(c,Ct[m[i]]) od; end;
```

$code := \mathbf{proc}(m, Ct)$ (9.2.2.5)

```
    local i, c;
    c := "";
    for i to length(m) do
        c := cat(c, Ct[m[i]])
    end do
end proc
```

```
> c:=code("ad emi",Ct);
```

$c := "10111000111010100000010001110111000101000"$ (9.2.2.6)

9.2.3. Prefix, szuffix, infix.

```
> IsPrefix("a","abc"); IsPrefix("","abc"); IsPrefix("abc",
  "abc");
```

true

true

true (9.2.3.1)

```
> IsSuffix("c","abc"); IsSuffix("b","abc");
```

true

false (9.2.3.2)

```
> HammingSearch("xy","axybcxyde",0); HammingSearch("xyz",
  "axybcxyzde",0);
  HammingSearch("xyz","axybcxyzde",1);
```

2

6

2 (9.2.3.3)

9.2.4. Kódfa.

9.2.5. Prefix kód, egyenletes kód, vesszős kód.

```
> isprefixcode:=proc(A,Ct) local f,i,j; f:=true;
  for i to length(A) do for j from i+1 to length(A) do
    if IsPrefix(Ct[A[i]],Ct[A[j]]) or IsPrefix(Ct[A[j]],Ct[A
  [i]])
    then f:=false fi;
  od; od; f; end;
```

$isprefixcode := \mathbf{proc}(A, Ct)$ (9.2.5.1)

```
    local f, i, j;
    f := true;
    for i to length(A) do
        for j from i + 1 to length(A) do
            if StringTools:-IsPrefix(Ct[A[i]],
            Ct[A[j]]) or StringTools:-IsPrefix(Ct[A[j]], Ct[A[i]]) then
                f := false
            end if
        end do
    end do;
    f
end proc
```

```
> isprefixcode(A,Ct);
```

$$true \tag{9.2.5.2}$$

```
> decodewithcodetable:=proc(c,A,Ct) local i,j,a,m,cc; m:="";
  cc:="";
  for i while i<=length(c) do cc:=cat(cc,c[i]);
    for j to length(A) do a:=A[j];
      if Ct[a]=cc then m:=cat(m,a); cc:=""; break fi; od;
  od; if length(cc)>0 then FAIL else m fi; end;
```

$$decodewithcodetable := \mathbf{proc}(c, A, Ct) \tag{9.2.5.3}$$

```
    local i, j, a, m, cc;
    m := "";
    cc := "";
    for i while i <= length(c) do
        cc := cat(cc, c[i]);
        for j to length(A) do
            a := A[j];
            if Ct[a] = cc then
                m := cat(m, a);
                cc := "";
                break
            end if
        end do
    end do;
    if 0 < length(cc) then
        FAIL
    else
        m
```

```
    end if
end proc
```

```
> decodewithcodetable(c,A,Ct);
```

"ad emi" (9.2.5.4)

```
> makedecodetable:=proc(A,Ct,Dt) local i;
  for i to length(A) do Dt[Ct[A[i]]]:=A[i] od; end;
```

$$makedecodetable := \mathbf{proc}(A, Ct, Dt) \quad (9.2.5.5)$$

```
    local i;
    for i to length(A) do
        Dt[Ct[A[i]]] := A[i]
    end do
end proc
```

```
> Dt:=table(): makedecodetable(A,Ct,Dt): print(Dt);
```

$$table([\text{"11101110101011110111000"} = ",", \text{"11101110101000"} = \text{"z"}, \quad (9.2.5.6)$$
"11101110111010101000" = ":", "1110101000" = "d",
"10111010111010111000" = ".", "111010101000" = "b",
"10101110111010101000" = "?", "111000" = "t",
"11101011101000" = "c", "111010111000" = "k",
"11101110101010101000" = "7", "1110111000" = "m",
"101010111000" = "v", "1110111011101010101000" = "8",
"101110111000" = "w", "10101010111000" = "4",
"11101110111011101000" = "9", "10111011101110111000" = "j",
"1010101000" = "h", "10101000" = "s",
"11101110111011101110111000" = "0", "11101011101110111000" = "y",
"11101110111000" = "o", "000" = " ", "1000" = "e",
"1010111000" = "u", "111011101000" = "g",
"11101110101011101110111000" = "q", "1011101110111011101110111000" = "1",
"101010101000" = "5", "11101010111000" = "x",
"11101000" = "n", "10101011101110111000" = "3",
"11101010101000" = "6", "10111000" = "a",
"101110101000" = "l", "101000" = "i",
"10101110111011101110111000" = "2", "10111011101000" = "p",
"11101010101011101110111000" = "-", "101011101000" = "f",
"1011101000" = "r"])

```
> decode:=proc(c::string,Dt::table) local i,m,cc; m:=""; cc:=
  "";
  for i while i<=length(c) do cc:=cat(cc,c[i]);
    if type(Dt[cc],string) then m:=cat(m,Dt[cc]); cc:=""; fi;
  od; if length(cc)>0 then FAIL else m fi; end;
```

(9.2.5.7)

```
decode := proc(c::string, Dt::table)
    local i, m, cc;
    m := "";
    cc := "";
    for i while i <= length(c) do
        cc := cat(cc, c[i]);
        if type(Dt[cc], string) then
            m := cat(m, Dt[cc]);
            cc := ""
        end if
    end do;
    if 0 < length(cc) then
        FAIL
    else
        m
    end if
end proc
```

(9.2.5.7)

```
> decode(c,Dt);
```

"ad emi" (9.2.5.8)

9.2.6. Példák.

->9.2.7. Feladat.

9.2.8. Feladat.

9.2.9. Tétel: McMillan-egyenlőtlenség.

```
> mcmillan:=proc(L::list(posint),r::posint)
  convert(map(x->r^(-x),L),`+`) end;
  mcmillan([6,7,5],2);
```

```
mcmillan := proc(L::(list(posint)), r::posint)
    convert(map(proc(x)
        option operator, arrow;
        r^(-x)
    end proc, L), +)
end proc
```

$$\frac{7}{128}$$ (9.2.9.1)

```
> shannoncode:=proc(B::string,L::list(posint)) local i,s,c,r,
  C,LL;
  r:=length(B); s:=0; C:=[];
```

```
LL:=[[L[i],i]$i=1..nops(L)]; LL:=sort(%,(x,y)->x[1]<y[1]);
for i to nops(LL) do
  c:=convert(s*r^LL[i][1],base,r); c:=map(x->B[x+1],c); c:=
cat("",op(c));
  c:=Reverse(c); c:=cat(Fill(B[1],LL[i][1]-length(c)),c);
  C:=[op(C),[c,LL[i][2]]]; s:=s+1/r^LL[i][1];
od; C:=sort(C,(x,y)->x[2]<y[2]); map(x->x[1],C); end;
```

```
shannoncode := proc(B::string, L::(list(posint)))
    local i, s, c, r, C, LL;
    r := length(B);
    s := 0;
    C := [];
    LL := [$([L[i], i], i = 1 ..nops(L))];
    LL := sort(%, proc(x, y)
        option operator, arrow;
        x[1] < y[1]
    end proc);
    for i to nops(LL) do
        c := convert(s*r^LL[i][1], base, r);
        c := map(proc(x)
            option operator, arrow;
            B[x+1]
        end proc, c);
        c := cat("", op(c));
        c := StringTools:-Reverse(c);
        c := cat(StringTools:-Fill(B[1], LL[i][1] - length(c)), c);
        C := [op(C), [c, LL[i][2]]];
        s := s + 1 / r^LL[i][1]
    end do;
    C := sort(C, proc(x, y)
        option operator, arrow;
        x[2] < y[2]
    end proc);
    map(proc(x)
        option operator, arrow;
        x[1]
    end proc, C)
end proc
```

(9.2.9.2)

```
> L:=[6,7,5]; B:="ab"; shannoncode(B,L);
```

$$L := [6, 7, 5]$$

$$B := \text{"ab"}$$

$$[\text{"aaaaba"}, \text{"aaaabba"}, \text{"aaaaa"}] \qquad (9.2.9.3)$$

▶ *->9.2.10. Feladat.*

▼ *->9.2.11. Feladat.*

▼ *->9.2.12. Feladat.*

▼ *9.2.13. Feladat.*

▼ *9.2.14. Átlagos szóhosszúság, optimális kód.*

```
> meanlength:=proc(F::list(nonnegative),L::list(posint))
  local i,s,m; if nops(F)<>nops(L) then return FAIL fi; s:=0;
  m:=0;
  for i to nops(F) do s:=s+F[i] od; if s=0 then return FAIL
  fi;
  for i to nops(F) do m:=m+L[i]*F[i] od; m/s; end;
```

$$meanlength := \textbf{proc}(F::(list(nonnegative)), L::(list(posint)))$$

$$\quad \textbf{local}\ i, s, m;$$

$$\quad \textbf{if}\ nops(F) <> nops(L)\ \textbf{then}$$

$$\quad\quad \textbf{return}\ FAIL$$

$$\quad \textbf{end if};$$

$$\quad s := 0;$$

$$\quad m := 0;$$

$$\quad \textbf{for}\ i\ \textbf{to}\ nops(F)\ \textbf{do}$$

$$\quad\quad s := s + F[i]$$

$$\quad \textbf{end do};$$

$$\quad \textbf{if}\ s = 0\ \textbf{then}$$

$$\quad\quad \textbf{return}\ FAIL$$

$$\quad \textbf{end if};$$

$$\quad \textbf{for}\ i\ \textbf{to}\ nops(F)\ \textbf{do}$$

$$\quad\quad m := m + L[i] * F[i]$$

$$\quad \textbf{end do};$$

$$\quad m / s$$

$$\textbf{end proc} \qquad (9.2.14.1)$$

```
> meanlength([1,1,2],[6,7,5]);
```

$$\frac{23}{4} \qquad (9.2.14.2)$$

▶ *9.2.15. Shannon tétele zajmentes csatornára.*

▼ *9.2.16. Tétel: Shannon-kód létezése.*

```
> shannonlength:=proc(F::list(positive),r::posint)
  local i,j,s,L; s:=0; L:=[]; if r<=1 then return FAIL fi;
  for i to nops(F) do s:=s+F[i] od;
  for i to nops(F) do for j while r^j<s/F[i] do od; L:=[op(L)
  ,j];
  od; L; end;
```

```
shannonlength := proc(F::(list(positive)), r::posint)
    local i, j, s, L;
    s := 0;
    L := [];
    if r <= 1 then
        return FAIL
    end if;
    for i to nops(F) do
        s := s + F[i]
    end do;
    for i to nops(F) do
        for j while r^j < s / F[i] do

        end do;
        L := [op(L), j]
    end do;
    L
end proc
```

(9.2.16.1)

```
> F:=[1,1,2,2]; L:=shannonlength([1,1,2,2],4); shannoncode
  ("0123",%);
```

$$F := [1, 1, 2, 2]$$

$$L := [2, 2, 1, 1]$$

$$["21", "20", "1", "0"]$$

(9.2.16.2)

```
> H(F,4); evalf(%); meanlength(F,L); evalf(%);
```

$$\frac{1}{6}\,\frac{\ln(6)}{\ln(2)} + \frac{1}{3}\,\frac{\ln(3)}{\ln(2)}$$

$$0.9591479170$$

$$\frac{4}{3}$$

$$1.333333333$$

(9.2.16.3)

▶ *9.2.17. Tétel: optimális kód konstrukciója.*

▼ *9.2.18. Huffmann-kód.*

```
> huffmannlength:=proc(F::list(nonnegative),r::posint)
  local i,l,h,rr,s,L,HH; if r<=1 then return FAIL fi;
  if nops(F)<=r then return [1$i=1..nops(F)] fi;
  L:=[[F[i],0,i]$i=1..nops(F)]; HH:=heap[new]((x,y)->x[1]>=y
  [1],op(L));
  rr:=2+((nops(L)-2) mod (r-1));
  do
    h:=heap[extract](HH);
    if heap[empty](HH) then break fi;
    s:=h[1]; L:=[h];
    for i from 2 to rr do h:=heap[extract](HH); L:=[op(L),h];
  s:=s+h[1]; od;
    heap[insert]([s,1,L],HH); rr:=r;
  od;
  HH:=heap[new]((x,y)->x[1]<=y[1],[h[2],0,h[3]]);
  do
    h:=heap[extract](HH); if h[1]=0 then heap[insert](h,HH);
  break fi;
    L:=h[3]; l:=h[2]+1;
    for i to nops(L) do heap[insert]([L[i][2],l,L[i][3]],HH);
  od;
  od; L:=[];
  while not heap[empty](HH) do
    h:=heap[extract](HH); L:=[op(L),[h[2],h[3]]];
  od; sort(L,(x,y)->x[2]<y[2]); map(x->x[1],%); end;
```

$$huffmannlength := \mathbf{proc}(F::(list(nonnegative)),\ r::posint) \tag{9.2.18.1}$$

local *i*, *l*, *h*, *rr*, *s*, *L*, *HH*;

if $r <= 1$ **then**

return *FAIL*

end if;

if $nops(F) <= r$ **then**

return $[\$(1,\ i = 1..nops(F))]$

end if;

$L := [\$([F[i],\ 0,\ i],\ i = 1..nops(F))]$;

$HH := heap[new](\mathbf{proc}(x,\ y)$

option *operator*, *arrow*;

$y[1] <= x[1]$

end proc, $op(L)$);

$rr := 2 + (mod(nops(L) - 2,\ r - 1))$;

do

```
        h := heap[extract](HH);
        if heap[empty](HH) then
            break
        end if;
        s := h[1];
        L := [h];
        for i from 2 to rr do
            h := heap[extract](HH);
            L := [op(L), h];
            s := s + h[1]
        end do;
        heap[insert]([s, 1, L], HH);
        rr := r
    end do;
    HH := heap[new](proc(x, y)
        option operator, arrow;
        x[1] <= y[1]
    end proc, [h[2], 0, h[3]]);
    do
        h := heap[extract](HH);
        if h[1] = 0 then
            heap[insert](h, HH);
            break
        end if;
        L := h[3];
        l := h[2] + 1;
        for i to nops(L) do
            heap[insert]([L[i][2], l, L[i][3]], HH)
        end do
    end do;
    L := [];
    while not heap[empty](HH) do
        h := heap[extract](HH);
        L := [op(L), [h[2], h[3]]]
    end do;
    sort(L, proc(x, y)
        option operator, arrow;
        x[2] < y[2]
```

```
    end proc);
    map(proc(x)
        option operator, arrow;
        x[1]
    end proc, %)
end proc
```

```
> F:=[1,1,2,2,3]; L:=huffmannlength(F,2);
  evalf(H(F,2)); evalf(meanlength(F,L));
```

$$F := [1, 1, 2, 2, 3]$$

$$L := [3, 3, 2, 2, 2]$$

$$2.197159724$$

$$2.222222222 \qquad (9.2.18.2)$$

```
> huffmanncode:=proc(B::string,L::list(posint)) local i,j,l,
  ll,b,c,LL,C;
  LL:=[[L[i],i]$i=1..nops(L)]; LL:=sort(LL,(x,y)->x[1]<y[1]);
  ll:=1; b[ll]:=0; C:=[];
  for i to nops(LL) do
    l:=ll; b[l]:=b[l]+1;
    while b[l]>length(B) do b[l]:=1; l:=l-1; b[l]:=b[l]+1;
  od;
    l:=LL[i][1]; while ll<l do ll:=ll+1; b[ll]:=1; od;
    c:=""; for j to l do c:=cat(c,B[b[j]]) od; C:=[op(C),[c,
  LL[i][2]]];
  od; sort(C,(x,y)->x[2]<y[2]); map(x->x[1],%); end;
```

$$huffmanncode := \mathbf{proc}(B::string, L::(list(posint))) \qquad (9.2.18.3)$$

```
    local i, j, l, ll, b, c, LL, C;
    LL := [ $([L[i], i], i = 1 ..nops(L))];
    LL := sort(LL, proc(x, y)
        option operator, arrow;
        x[1] < y[1]
    end proc);
    ll := 1;
    b[ll] := 0;
    C := [ ];
    for i to nops(LL) do
        l := ll;
        b[l] := b[l] + 1;
        while length(B) < b[l] do
            b[l] := 1;
            l := l - 1;
            b[l] := b[l] + 1
```

```
        end do;
        l := LL[i][1];
        while ll < l do
            ll := ll + 1;
            b[ll] := 1
        end do;
        c := "";
        for j to l do
            c := cat(c, B[b[j]])
        end do;
        C := [op(C), [c, LL[i][2]]]
    end do;
    sort(C, proc(x, y)
        option operator, arrow;
        x[2] < y[2]
    end proc);
    map(proc(x)
        option operator, arrow;
        x[1]
    end proc, %)
end proc
```

```
> huffmanncode("ab",L);
```

["bbb", "bba", "ba", "ab", "aa"] (9.2.18.4)

9.2.19. Példa.

```
> F:=[17,2,13,2,1,31,2,17,6,9]; L:=huffmannlength(F,3);
  evalf(H(F,3)); evalf(meanlength(F,L)); huffmanncode("012",
  L);
```

$$F := [17, 2, 13, 2, 1, 31, 2, 17, 6, 9]$$

$$L := [2, 3, 2, 3, 4, 1, 4, 2, 2, 2]$$

1.726772590

1.790000000

["21", "221", "20", "220", "2221", "0", "2220", "12", "11", "10"] (9.2.19.1)

```
> L:=shannonlength(F,3); shannoncode("012",L); evalf
  (meanlength(F,L));
```

$$L := [2, 4, 2, 4, 5, 2, 4, 2, 3, 3]$$

["10", "1122", "02", "1121", "12000", "01", "1120", "00", "111", "110"]

(9.2.19.2)

2.300000000 (9.2.19.2)

```
> F:=[1247,190,65,197,1407,78,326,165,444,105,541,605,352,
  638,683,96,0,367,600,761,226,199,9,1,261,437]; L:=
  huffmannlength(F,2);
  huffmanncode("01",L); evalf(meanlength(F,L));
```

$F := [1247, 190, 65, 197, 1407, 78, 326, 165, 444, 105, 541, 605, 352, 638, 683, 96, 0, 367, 600, 761, 226, 199, 9, 1, 261, 437]$

$L := [3, 6, 8, 6, 3, 7, 5, 6, 4, 7, 4, 4, 5, 4, 4, 7, 10, 5, 4, 4, 5, 6, 9, 10, 5, 4]$

["001", "111101", "11111110", "111100", "000", "1111110", "11100", "111011", "1011", "1111101", "1010", "1001", "11011", "1000", "0111", "1111100", "1111111111", "11010", "0110", "0101", "11001", "111010", "111111110", "1111111110", "11000", "0100"]

4.152800000 (9.2.19.3)

```
> F:=map(x->x+1,F); L:=shannonlength(F,2); shannoncode("01",
  L);
  evalf(meanlength(F,L)); evalf(H(F,2));
```

$F := [1248, 191, 66, 198, 1408, 79, 327, 166, 445, 106, 542, 606, 353, 639, 684, 97, 1, 368, 601, 762, 227, 200, 10, 2, 262, 438]$

$L := [4, 6, 8, 6, 3, 7, 5, 6, 5, 7, 5, 5, 5, 4, 4, 7, 14, 5, 5, 4, 6, 6, 10, 13, 6, 5]$

["0101", "101101", "10111110", "101100", "000", "1011110", "10011", "101011", "10010", "1011101", "10001", "10000", "01111", "0100", "0011", "1011100", "10111111010010", "01110", "01101", "0010", "101010", "101001", "1011111100", "1011111101000", "101000", "01100"]

4.594254937

4.133367920 (9.2.19.4)

▼ ***9.2.20. Adaptív Huffmann-kód.***

▼ ***->9.2.21. Feladat.***

▼ ***9.2.22. Feladat.***

▼ ***->9.2.23. Feladat.***

▼ ****9.2.24. Feladat.***

▼ ***9.2.25. Feladat.***

▼ ****9.2.26. Feladat.***

▼ ***9.2.27. Feladat.***

▼ *9.2.28. Feladat.
▼ 9.2.29. Feladat.
▼ *9.2.30. Feladat.
▶ 9.2.31. A kódolandó ábécé kiterjesztése.
▼ ->9.2.32. Feladat.
▼ *9.2.33. Feladat.
▶ 9.2.34. Szótárkódok.
▼ *9.2.35. Futamkódolás.
▼ *9.2.36. LZ77.
▼ *9.2.37. A gzip parancs.
▼ *9.2.38. LZW-kódok.

```
> LZW:=proc(m,T,N) local i,n,alpha,beta,gamma,L; n:=128; L:=
  [];
  for i from 0 to n-1 do
    T[convert([i],bytes)]:=cat("",convert(i,hex));
  od;
  if length(m)=0 then return L fi; alpha:=m[1];
  for i from 2 to length(m) do
    beta:=m[i]; gamma:=cat(alpha,beta);
    if type(T[gamma],string) then alpha:=gamma; next fi;
    if n<N then T[gamma]:=cat("",convert(n,hex)); n:=n+1; fi;
    L:=[op(L),T[alpha]]; alpha:=beta;
  od; L:=[op(L),T[alpha]]; end;
```

$LZW := \mathbf{proc}(m, T, N)$ (9.2.38.1)

$\mathbf{local}\ i, n, \alpha, \beta, \gamma, L;$
$n := 128;$
$L := [\,];$
$\mathbf{for}\ i\ \mathbf{from}\ 0\ \mathbf{to}\ n-1\ \mathbf{do}$
$T[\,convert([\,i], bytes)\,] := cat("", convert(i, hex))$
$\mathbf{end\ do};$
$\mathbf{if}\ length(m) = 0\ \mathbf{then}$
$\mathbf{return}\ L$
$\mathbf{end\ if};$
$\alpha := m[\,1\,];$
$\mathbf{for}\ i\ \mathbf{from}\ 2\ \mathbf{to}\ length(m)\ \mathbf{do}$
$\beta := m[\,i\,];$
$\gamma := cat(\alpha, \beta);$
$\mathbf{if}\ type(T[\,\gamma\,], string)\ \mathbf{then}$

```
            α := γ;
            next
        end if;
        if n < N then
            T[γ] := cat("", convert(n, hex));
            n := n + 1
        end if;
        L := [op(L), T[α]];
        α := β
    end do;
    L := [op(L), T[α]]
end proc
```

```
> m:="GUGOGOGOGOL"; Ct:=table(); L:=LZW(m,Ct,256);
```

$$m := \text{"GUGOGOGOGOL"}$$

$$Ct := table([\,])$$

$$L := [\text{"47"}, \text{"55"}, \text{"47"}, \text{"4F"}, \text{"82"}, \text{"84"}, \text{"4F"}, \text{"4C"}] \qquad (9.2.38.2)$$

```
> iLZW:=proc(L,T,N) local i,j,m,n,alpha,beta,gamma; n:=128;
  m:="";
  for i from 0 to n-1 do
    T[cat("",convert(i,hex))]:=convert([i],bytes);
  od;
  if nops(L)=0 then return m fi; j:=L[1]; alpha:=T[j];
  for i from 2 to nops(L) do
    j:=L[i];
    if type(T[j],string) then
      beta:=T[j]; gamma:=beta[1];
      if n<N then T[cat("",convert(n,hex))]:=cat(alpha,gamma)
  ; n:=n+1; fi;
      m:=cat(m,alpha); alpha:=beta;
    else
      gamma:=alpha[1]; beta:=cat(alpha,gamma);
      T[cat("",convert(n,hex))]:=beta; n:=n+1;
      m:=cat(m,alpha); alpha:=beta;
    fi;
  od; m:=cat(m,alpha); end;
```

$$iLZW := \mathbf{proc}(L, T, N) \qquad (9.2.38.3)$$

```
    local i, j, m, n, α, β, γ;
    n := 128;
    m := "";
    for i from 0 to n − 1 do
        T[cat("", convert(i, hex))] := convert([i], bytes)
```

```
    end do;
    if nops(L) = 0 then
        return m
    end if;
    j := L[1];
    α := T[j];
    for i from 2 to nops(L) do
        j := L[i];
        if type(T[j], string) then
            β := T[j];
            γ := β[1];
            if n < N then
                T[cat("", convert(n, hex))] := cat(α, γ);
                n := n + 1
            end if;
            m := cat(m, α);
            α := β
        else
            γ := α[1];
            β := cat(α, γ);
            T[cat("", convert(n, hex))] := β;
            n := n + 1;
            m := cat(m, α);
            α := β
        end if
    end do;
    m := cat(m, α)
end proc
```

```
> Dt:=table(); iLZW(L,Ct,256);
```

$$Dt := table([\,])$$

"GUGOGOGOGOL" (9.2.38.4)

*9.2.39. Példa az LZW-kódra.

*9.2.40. A compress parancs.

*9.2.41. Digitalizálás.

*9.2.42. DFT és IDFT.

```
> with(DiscreteTransforms);
```

$$[FourierTransform, InverseFourierTransform] \tag{9.2.42.1}$$

```
> Z:=Vector(5,[0,1+I,0,0,0]);
```

$$Z := \begin{bmatrix} 0 \\ 1+I \\ 0 \\ 0 \\ 0 \end{bmatrix} \tag{9.2.42.2}$$

```
> ZZ:=FourierTransform(Z);
```

$$ZZ := \begin{bmatrix} 0.4472135954999579\,28 + 0.4472135954999579\,28\,I \\ 0.5635220053010304\,71 - 0.2871288030510094\,63\,I \\ -0.0989378428154226858 - 0.6246689549345563\,20\,I \\ -0.6246689549345563\,20 - 0.0989378428154226858\,I \\ -0.2871288030510094\,63 + 0.5635220053010304\,71\,I \end{bmatrix} \tag{9.2.42.3}$$

```
> InverseFourierTransform(ZZ);
```

$$\begin{bmatrix} -4.965068306494545\,86\,10^{-17} - 4.965068306494545\,86\,10^{-17}\,I \\ 1. + 1.\,I \\ -1.078835642768590\,31\,10^{-18} - 1.078835642768590\,31\,10^{-18}\,I \\ -3.163776435534854\,58\,10^{-18} - 3.163776435534854\,58\,10^{-18}\,I \\ 7.757919228897727\,92\,10^{-19} + 7.757919228897727\,92\,10^{-19}\,I \end{bmatrix} \tag{9.2.42.4}$$

```
> FourierTransform(ZZ);
```

$$\begin{bmatrix} -4.965068306494545\,86\,10^{-17} - 4.965068306494545\,86\,10^{-17}\,I \\ 7.757919228897727\,92\,10^{-19} + 7.757919228897727\,92\,10^{-19}\,I \\ -3.163776435534854\,58\,10^{-18} - 3.163776435534854\,58\,10^{-18}\,I \\ -1.078835642768590\,31\,10^{-18} - 1.078835642768590\,31\,10^{-18}\,I \\ 1. + 1.\,I \end{bmatrix} \tag{9.2.42.5}$$

*9.2.43. FFT.

*9.2.44. FFT algoritmus.

```
> #
  # This procedure do the bit reversation of the
  # number x which is k bit long.
  #
```

```
reverse:=proc(x,k) local xx,i,y;
  xx:=x; y:=0;
  for i to k do
    if type(xx,odd) then
      y:=2*y+1; xx:=(xx-1)/2;
    else
      y:=2*y; xx:=xx/2;
    fi;
  od; y; end;
```

```
reverse := proc(x, k)                                   (9.2.44.1)
    local xx, i, y;
    xx := x;
    y := 0;
    for i to k do
        if type(xx, odd) then
            y := 2 * y + 1;
            xx := 1 / 2 * xx - 1 / 2
        else
            y := 2 * y;
            xx := 1 / 2 * xx
        end if
    end do;
    y
end proc
```

```
> #
  # This procedure do the butterfly operation
  # between the two terms A[i] and A[j] of the
  # table A. The multiplier is w.
  #

  butterfly:=proc(A,i,j,w) local X,Y;
    X:=A[i]; Y:=A[j]*w; A[i]:=X+Y; A[j]:=X-Y;
  end;
```

```
butterfly := proc(A, i, j, w)                           (9.2.44.2)
    local X, Y;
    X := A[i];
    Y := A[j] * w;
    A[i] := X + Y;
    A[j] := X - Y
end proc
```

```
> #
```

```
# This is an FFT procedure.
# It use the butterfly procedure to operate on the vector
A.
# The number of rounds is k in the FFT.
# T is a table of the powers of primitive root of unity.
#

fft:=proc(A,T,k) local l,s,w,t;
  for l from 0 to k-1 do
    for s from 0 to 2^l-1 do
      w:=T[s];
      for t from 0 to 2^(k-l-1)-1 do
        butterfly(A,2^(k-l)*s+t,2^(k-l)*s+t+2^(k-l-1),w);
      od;
    od;
  od;
end;
```

$$fft := \mathbf{proc}(A, T, k) \tag{9.2.44.3}$$

```
 local l, s, w, t;
 for l from 0 to k - 1 do
     for s from 0 to 2^l - 1 do
         w := T[s];
         for t from 0 to 2^(k - l - 1) - 1 do
             butterfly(A, 2^(k - l) * s + t, 2^(k - l) * s + t
             + 2^(k - l - 1), w)
         end do
     end do
 end do
end proc
```

```
> #
  # The pre procedure do the preparation for FFT:
  # The table T[0..2^(k-1)] of powers of the primitive
  # root of unity prepared.
  #

  pre:=proc(T,k) local i;
  Digits:=2*Digits;
  for i from 0 to 2^k-1 do
    T[i]:=evalf(exp(-2*Pi*I*reverse(i,k)/2^(k+1)));
  od;
  Digits:=Digits/2;
  end;
```

$$pre := \mathbf{proc}(T, k) \tag{9.2.44.4}$$

```
 local i;
```

```
    Digits := 2 * Digits;
    for i from 0 to 2^k - 1 do
        T[i] := evalf(exp(-2 * I * π * reverse(i, k) / 2^(k+1)))
    end do;
    Digits := 1 / 2 * Digits
end proc
```

```
> k:=5; A:=array(0..2^k-1); for i from 0 to 2^k -1 do A[i]:=0
  od:
  T:=array(0..2^(k-1)-1);
```

$$k := 5$$

$$A := array(0..31, [\,])$$

$$T := array(0..15, [\,]) \tag{9.2.44.5}$$

```
> pre(T,k-1): A[1]:=1+I: print(A);
```

$$ARRAY([0..31], [0 = 0, 1 = 1 + I, 2 = 0, 3 = 0, 4 = 0, 5 = 0, 6 = 0, 7 = 0, 8 = 0, 9 = 0, 10 = 0, 11 = 0, 12 = 0, 13 = 0, 14 = 0, 15 = 0, 16 = 0, 17 = 0, 18 = 0, 19 = 0, 20 = 0, 21 = 0, 22 = 0, 23 = 0, 24 = 0, 25 = 0, 26 = 0, 27 = 0, 28 = 0, 29 = 0, 30 = 0, 31 = 0]) \tag{9.2.44.6}$$

```
> fft(A,T,k): print(A);
```

$$ARRAY([0..31], [0 = 1. + 1.\,I, 1 = -1. - 1.\,I, 2 = 1. - 1.\,I, 3 = -1. + 1.\,I, 4 = 1.414213562 + 0.\,I, 5 = -1.414213562 - 0.\,I, 6 = 0. - 1.414213562\,I, 7 = 0. + 1.414213562\,I, 8 = 1.306562965 + 0.5411961001\,I, 9 = -1.306562965 - 0.5411961001\,I, 10 = 0.5411961001 - 1.306562965\,I, 11 = -0.5411961001 + 1.306562965\,I, 12 = 1.306562965 - 0.5411961001\,I, 13 = -1.306562965 + 0.5411961001\,I, 14 = -0.5411961001 - 1.306562965\,I, 15 = 0.5411961001 + 1.306562965\,I, 16 = 1.175875602 + 0.7856949584\,I, 17 = -1.175875602 - 0.7856949584\,I, 18 = 0.7856949584 - 1.175875602\,I, 19 = -0.7856949584 + 1.175875602\,I, 20 = 1.387039845 - 0.2758993793\,I, 21 = -1.387039845 + 0.2758993793\,I, 22 = -0.2758993793 - 1.387039845\,I, 23 = 0.2758993793 + 1.387039845\,I, 24 = 1.387039845 + 0.2758993793\,I, 25 = -1.387039845 - 0.2758993793\,I, 26 = 0.2758993793 - 1.387039845\,I, 27 = -0.2758993793 + 1.387039845\,I, 28 = 1.175875602 - 0.7856949584\,I, 29 \tag{9.2.44.7}$$

$= -1.175875602$

$+ 0.7856949584\,I,\ 30 = -0.7856949584 - 1.175875602\,I,\ 31$

$= 0.7856949584 + 1.175875602\,I]\big)$

```
> for i from 0 to 2^k-1 do
    j:=reverse(i,k);
    if i<j then x:=A[i]; A[i]:=A[j]; A[j]:=x; fi;
  od: print(A);
```

$ARRAY\big([0..31], [0 = 1. + 1.\,I,\ 1 = 1.175875602$ (9.2.44.8)

$+ 0.7856949584\,I,\ 2 = 1.306562965$

$+ 0.5411961001\,I,\ 3 = 1.387039845$

$+ 0.2758993793\,I,\ 4 = 1.414213562$

$+ 0.\,I,\ 5 = 1.387039845 - 0.2758993793\,I,\ 6$

$= 1.306562965 - 0.5411961001\,I,\ 7$

$= 1.175875602 - 0.7856949584\,I,\ 8 = 1. - 1.\,I,\ 9$

$= 0.7856949584 - 1.175875602\,I,\ 10$

$= 0.5411961001 - 1.306562965\,I,\ 11$

$= 0.2758993793 - 1.387039845\,I,\ 12 = 0. - 1.414213562\,I,\ 13$

$= -0.2758993793 - 1.387039845\,I,\ 14$

$= -0.5411961001 - 1.306562965\,I,\ 15$

$= -0.7856949584 - 1.175875602\,I,\ 16 = -1. - 1.\,I,\ 17$

$= -1.175875602 - 0.7856949584\,I,\ 18$

$= -1.306562965 - 0.5411961001\,I,\ 19$

$= -1.387039845 - 0.2758993793\,I,\ 20 = -1.414213562 - 0.\,I,\ 21$

$= -1.387039845 + 0.2758993793\,I,\ 22 = -1.306562965$

$+ 0.5411961001\,I,\ 23 = -1.175875602$

$+ 0.7856949584\,I,\ 24 = -1. + 1.\,I,\ 25 = -0.7856949584$

$+ 1.175875602\,I,\ 26 = -0.5411961001$

$+ 1.306562965\,I,\ 27 = -0.2758993793 + 1.387039845\,I,\ 28 = 0.$

$+ 1.414213562\,I,\ 29 = 0.2758993793$

$+ 1.387039845\,I,\ 30 = 0.5411961001$

$+ 1.306562965\,I,\ 31 = 0.7856949584 + 1.175875602\,I]\big)$

*9.2.45. IFFT algoritmus.

```
> #
  # This procedure do the inverse butterfly operation
  # between the two terms A[i] and A[j] of the
  # table A. The power of primitive unity is w.
  #
```

```
ibutterfly:=proc(A,i,j,w) local X,Y,e;
  X:=A[i]+A[j]; Y:=A[i]-A[j]; A[i]:=X; A[j]:=Y/w;
end;
```

ibutterfly := **proc**(*A*, *i*, *j*, *w*) (9.2.45.1)

local *X*, *Y*, *e*;
X := *A*[*i*] + *A*[*j*];
Y := *A*[*i*] − *A*[*j*];
A[*i*] := *X*;
A[*j*] := *Y* / *w*
end proc

```
> #
  # This procedure is a floating point IFFT procedure.
  # It use the ibutterfly procedure to operate on the vector
  A.
  # The number of round in the FFT is k and T is a table of
  the
  # powers of primitive root of unity.
  #

  ifft:=proc(A,T,k) local l,s,w,t;
  for l from k-1 to 0 by -1 do
    for s from 0 to 2^l-1 do
      w:=T[s];
      for t from 0 to 2^(k-l-1)-1 do
        ibutterfly(A,2^(k-l)*s+t,2^(k-l)*s+t+2^(k-l-1),w);
      od;
    od;
  od; end;
```

ifft := **proc**(*A*, *T*, *k*) (9.2.45.2)

local *l*, *s*, *w*, *t*;
for *l* **from** *k* − 1 **by** −1 **to** 0 **do**
for *s* **from** 0 **to** 2^*l* − 1 **do**
w := *T*[*s*];
for *t* **from** 0 **to** 2^(*k* − *l* − 1) − 1 **do**
ibutterfly(*A*, 2^(*k* − *l*) * *s* + *t*, 2^(*k* − *l*) * *s* + *t*
+ 2^(*k* − *l* − 1), *w*)
end do
end do
end do
end proc

```
> for i from 0 to 2^k-1 do
    j:=reverse(i,k);
    if i<j then x:=A[i]; A[i]:=A[j]; A[j]:=x; fi;
```

```
   od: print(A);
```

$$ARRAY([0..31], [0 = 1. + 1.\,I,\ 1 = -1. - 1.\,I,\ 2 = 1. - 1.\,I,\ 3 = -1. + 1.\,I,\ 4 = 1.414213562 + 0.\,I,\ 5 = -1.414213562 - 0.\,I,\ 6 = 0. - 1.414213562\,I,\ 7 = 0. + 1.414213562\,I,\ 8 = 1.306562965 + 0.5411961001\,I,\ 9 = -1.306562965 - 0.5411961001\,I,\ 10 = 0.5411961001 - 1.306562965\,I,\ 11 = -0.5411961001 + 1.306562965\,I,\ 12 = 1.306562965 - 0.5411961001\,I,\ 13 = -1.306562965 + 0.5411961001\,I,\ 14 = -0.5411961001 - 1.306562965\,I,\ 15 = 0.5411961001 + 1.306562965\,I,\ 16 = 1.175875602 + 0.7856949584\,I,\ 17 = -1.175875602 - 0.7856949584\,I,\ 18 = 0.7856949584 - 1.175875602\,I,\ 19 = -0.7856949584 + 1.175875602\,I,\ 20 = 1.387039845 - 0.2758993793\,I,\ 21 = -1.387039845 + 0.2758993793\,I,\ 22 = -0.2758993793 - 1.387039845\,I,\ 23 = 0.2758993793 + 1.387039845\,I,\ 24 = 1.387039845 + 0.2758993793\,I,\ 25 = -1.387039845 - 0.2758993793\,I,\ 26 = 0.2758993793 - 1.387039845\,I,\ 27 = -0.2758993793 + 1.387039845\,I,\ 28 = 1.175875602 - 0.7856949584\,I,\ 29 = -1.175875602 + 0.7856949584\,I,\ 30 = -0.7856949584 - 1.175875602\,I,\ 31 = 0.7856949584 + 1.175875602\,I]) \tag{9.2.45.3}$$

```
> ifft(A,T,k): print(A);
```

$$ARRAY([0..31], [0 = 0. + 0.\,I,\ 1 = 32.00000000 + 32.00000000\,I,\ 2 = 0. + 0.\,I,\ 3 = 0. + 0.\,I,\ 4 = 0. + 0.\,I,\ 5 = -8.28427124\,10^{-10} - 8.28427124\,10^{-10}\,I,\ 6 = 0. + 0.\,I,\ 7 = 0. + 0.\,I,\ 8 = 0. + 0.\,I,\ 9 = -4.000000000\,10^{-9} - 4.000000000\,10^{-9}\,I,\ 10 = 0. + 0.\,I,\ 11 = 0. + 0.\,I,\ 12 = 0. + 0.\,I,\ 13 = 2.\,10^{-9} + 2.\,10^{-9}\,I,\ 14 = 0. + 0.\,I,\ 15 = 0. + 0.\,I,\ 16 = 0. + 0.\,I,\ 17 = 0. + 0.\,I,\ 18 = 0. + 0.\,I,\ 19 = 0. + 0.\,I,\ 20 = 0. + 0.\,I,\ 21 = 4.828427124\,10^{-9} + 4.828427124\,10^{-9}\,I,\ 22 = 0. + 0.\,I,\ 23 = 0. + 0.\,I,\ 24 = 0. + 0.\,I,\ 25 = 0. + 0.\,I,\ 26 = 0. + 0.\,I,\ 27 = 0. + 0.\,I,\ 28 = 0. + 0.\,I,\ 29 = 2.\,10^{-9} + 2.\,10^{-9}\,I,\ 30 = 0. + 0.\,I,\ 31 = 0. + 0.\,I]) \tag{9.2.45.4}$$

*9.2.46. Gyors szorzás FFT-vel.

```
> #
  # The digmul procedure do the digit-by-digit
  # multiplication of the two numbers after the
  # fft's. The result will be in the first table.
  #

  digmul:=proc(A,B,k) local i;
  for i from 0 to 2^k-1 do A[i]:=A[i]*B[i]; od;
  end;
```

$$digmul := \mathbf{proc}(A, B, k) \quad (9.2.46.1)$$

```
    local i;
    for i from 0 to 2^k - 1 do
        A[i] := A[i] * B[i]
    end do
end proc
```

```
> A:=array(0..2^k-1); for i from 0 to 2^k -1 do A[i]:=0 od:
  B:=array(0..2^k-1); for i from 0 to 2^k -1 do B[i]:=0 od:
```

$$A := array(0..31, [\,])$$
$$B := array(0..31, [\,]) \quad (9.2.46.2)$$

```
> A[0]:=1: A[1]:=1: A[2]:=1: print(A);
```

$$ARRAY([0..31], [0 = 1, 1 = 1, 2 = 1, 3 = 0, 4 = 0, 5 = 0, 6 = 0, 7 = 0, 8 = 0, 9 = 0, 10 = 0, 11 = 0, 12 = 0, 13 = 0, 14 = 0, 15 = 0, 16 = 0, 17 = 0, 18 = 0, 19 = 0, 20 = 0, 21 = 0, 22 = 0, 23 = 0, 24 = 0, 25 = 0, 26 = 0, 27 = 0, 28 = 0, 29 = 0, 30 = 0, 31 = 0]) \quad (9.2.46.3)$$

```
> #
  # This is the polynom multiplication, do multiplication or
  squaring.
  # A and B are the two polynomials, the FFT and IFFT use k
  rounds.
  #

  polmulfft:=proc(A,B,k) global T;
  if A=B then
    fft(A,T,k);
    digmul(A,A,k);
    ifft(A,T,k);
  else
    fft(A,T,k);
    fft(B,T,k);
    digmul(A,B,k);
    ifft(A,T,k);
  fi; A; end;
```

$$polmulfft := \mathbf{proc}(A, B, k) \quad (9.2.46.4)$$

```
    global T;
    if A = B then
        fft(A, T, k);
        digmul(A, A, k);
        ifft(A, T, k)
    else
        fft(A, T, k);
        fft(B, T, k);
        digmul(A, B, k);
        ifft(A, T, k)
    end if;
    A
end proc
```

```
> polmulfft(A,A,k): print(map(x->x/2^k,A));
```

$$ARRAY([0..31], [0 = 0.9999999994 + 0.\,I,\ 1 = 1.999999999 + 0.\,I,\ 2 = 2.999999999 + 0.\,I,\ 3 = 2.000000000 + 0.\,I,\ 4 = 1.000000000 + 0.\,I,\ 5 = 8.080582619\ 10^{-11} + 0.\,I,\ 6 = -1.061335052\ 10^{-10} + 0.\,I,\ 7 = 2.251727930\ 10^{-10} + 0.\,I,\ 8 = 0. + 0.\,I,\ 9 = -1.875000000\ 10^{-10} + 0.\,I,\ 10 = -1.419231619\ 10^{-10} + 0.\,I,\ 11 = -6.243093422\ 10^{-10} + 0.\,I,\ 12 = -2.758883476\ 10^{-10} + 0.\,I,\ 13 = 1.325825215\ 10^{-10} + 0.\,I,\ 14 = 6.893616238\ 10^{-10} + 0.\,I,\ 15 = 5.787261838\ 10^{-10} + 0.\,I,\ 16 = 6.250000000\ 10^{-10} + 0.\,I,\ 17 = 6.250000000\ 10^{-10} + 0.\,I,\ 18 = 6.250000000\ 10^{-10} + 0.\,I,\ 19 = 3.125000000\ 10^{-10} + 0.\,I,\ 20 = 6.250000000\ 10^{-10} + 0.\,I,\ 21 = 1.691941738\ 10^{-10} + 0.\,I,\ 22 = 1.945218528\ 10^{-10} + 0.\,I,\ 23 = 7.660390225\ 10^{-11} + 0.\,I,\ 24 = 0. + 0.\,I,\ 25 = 1.875000000\ 10^{-10} + 0.\,I,\ 26 = 1.419231619\ 10^{-10} + 0.\,I,\ 27 = -6.906577500\ 10^{-13} + 0.\,I,\ 28 = -9.911165238\ 10^{-11} + 0.\,I,\ 29 = -1.325825215\ 10^{-10} + 0.\,I,\ 30 = -7.777499712\ 10^{-10} - 0.\,I,\ 31 = -6.305028788\ 10^{-10} - 0.\,I]) \tag{9.2.46.5}$$

```
> #
  # The fftpre procedure do the preparation for the
  # table for fft, where x is the number, A is the
  # table, m the modulus, and k is the number of
  # rounds for the fft, hence the table is 2^k long.
  #
```

```
fftpre:=proc(x,A,m,k) local i,xx;
xx:=x;
for i from 0 to 2^k-1 do A[i]:=irem(xx,m,'xx'); od;
end;
```

$$fftpre := \mathbf{proc}(x, A, m, k) \quad (9.2.46.6)$$

```
 local i, xx;
 xx := x;
 for i from 0 to 2^k - 1 do
     A[i] := irem(xx, m, 'xx')
 end do
end proc
```

```
> #
  # The norm procedure do the normalization
  # after the ifft. A is the table, m the modulus,
  # and k is the number of rounds for the fft,
  # hence the table is 2^k long. The fraction parts are
  # left in the table A.
  #

  fftnorm:=proc(A,m,k) local i,x;
  x:=0;
  for i from 2^k-1 to 0 by -1 do
    A[i]:=A[i]/2^k;
    x:=m*x+round(A[i]);
    A[i]:=A[i]-round(A[i]);
  od; x; end;
```

$$fftnorm := \mathbf{proc}(A, m, k) \quad (9.2.46.7)$$

```
 local i, x;
 x := 0;
 for i from 2^k - 1 by -1 to 0 do
     A[i] := A[i] / 2^k;
     x := m*x + round(A[i]);
     A[i] := A[i] - round(A[i])
 end do;
 x
end proc
```

```
> #
  # This is the main procedure, do the multiplication or the
  squaring.
  # a and b are the two numbers, m is the modulus for the
  preparation.
  # The FFT and IFFT use k rounds.
```

```
#

mulfft:=proc(a,b,m,k) global A,B,T;
if a=b then
  fftpre(a,A,m,k);
  fft(A,T,k);
  digmul(A,A,k);
  ifft(A,T,k);
  fftnorm(A,m,k);
else
  fftpre(a,A,m,k);
  fft(A,T,k);
  fftpre(b,B,m,k);
  fft(B,T,k);
  digmul(A,B,k);
  ifft(A,T,k);
  fftnorm(A,m,k);
fi;
end;
```

$$mulfft := \mathbf{proc}(a, b, m, k) \tag{9.2.46.8}$$

global $A, B, T;$
if $a = b$ **then**
$fftpre(a, A, m, k);$
$fft(A, T, k);$
$digmul(A, A, k);$
$ifft(A, T, k);$
$fftnorm(A, m, k)$
else
$fftpre(a, A, m, k);$
$fft(A, T, k);$
$fftpre(b, B, m, k);$
$fft(B, T, k);$
$digmul(A, B, k);$
$ifft(A, T, k);$
$fftnorm(A, m, k)$
end if
end proc

```
> mulfft(123456789,987654321,20,5);
```

$$121932631112635269 \tag{9.2.46.9}$$

```
> 123456789*987654321;
```

$$121932631112635269 \tag{9.2.46.10}$$

```
> print(A);
```

(9.2.46.11)

$$ARRAY([0..31], [0 = 6.2\,10^{-8} + 0.\,I,\ 1 = -1.\,10^{-7} + 0.\,I,\ 2 = 0. + 0.\,I,\ 3 = 0. + 0.\,I,\ 4 = 0. + 0.\,I,\ 5 = 0. + 0.\,I,\ 6 = 0. + 0.\,I,\ 7 = 0. + 0.\,I,\ 8 = 0. + 0.\,I,\ 9 = 0. + 0.\,I,\ 10 = 0. + 0.\,I,\ 11 = 1.\,10^{-7} + 0.\,I,\ 12 = 7.\,10^{-8} + 0.\,I,\ 13 = -1.830582619\,10^{-8} + 0.\,I,\ 14 = 0. + 0.\,I,\ 15 = 1.325825215\,10^{-7} + 0.\,I,\ 16 = -6.250000000\,10^{-8} + 0.\,I,\ 17 = 1.250000000\,10^{-7} + 0.\,I,\ 18 = -1.250000000\,10^{-7} + 0.\,I,\ 19 = 0. + 0.\,I,\ 20 = 0. + 0.\,I,\ 21 = 0. + 0.\,I,\ 22 = 0. + 0.\,I,\ 23 = 0. + 0.\,I,\ 24 = 0. + 0.\,I,\ 25 = -6.250000000\,10^{-8} + 0.\,I,\ 26 = 0. + 0.\,I,\ 27 = -6.250000000\,10^{-8} + 0.\,I,\ 28 = -6.875000000\,10^{-8} + 0.\,I,\ 29 = -1.066941738\,10^{-7} + 0.\,I,\ 30 = 0. - 0.\,I,\ 31 = -1.325825215\,10^{-7} - 0.\,I]) \tag{9.2.46.11}$$

*9.2.47. DCT és IDCT.

*9.2.48. Kétdimenziós DFT és DCT.

```
> #
  # This procedure is one round of an FFT procedure.
  # It use the butterfly procedure to operate on the
  # vector A. Round r is done, with siccor size s,
  # and repeated until the vector A is finished,
  # i.e. in size N. The T is a table of the powers of
  # primitive root of unity.
  #

  fftround:=proc(A,T,r,s,N) local i,j,w,t;
  i:=0;
  while i<N do
    for j from 0 to 2^r-1 do
      w:=T[j];
      for t from 0 to s-1 do
        butterfly(A,i+t,i+s+t,w);
      od;
      i:=i+2*s;
    od;
  od; end;
```

$$fftround := \mathbf{proc}(A, T, r, s, N) \tag{9.2.48.1}$$

$$\mathbf{local}\ i, j, w, t;$$

$$i := 0;$$

$$\mathbf{while}\ i < N\ \mathbf{do}$$

$$\mathbf{for}\ j\ \mathbf{from}\ 0\ \mathbf{to}\ 2\text{^}r - 1\ \mathbf{do}$$

$$w := T[j];$$

$$\mathbf{for}\ t\ \mathbf{from}\ 0\ \mathbf{to}\ s - 1\ \mathbf{do}$$

```
            butterfly(A, i+t, i+s+t, w)
        end do;
        i := i+2*s
    end do
  end do
end proc
```

```
> #
  # This procedure is an IFFT round. It use the ibutterfly
  # procedure to operate on the vector A. Round r is done
  # with siccor size s and repeated until the vector A is
  # finished, i.e. in size N. The T is a table of the
  # powers of primitive root of unity.
  #

  ifftround:=proc(A,T,r,s,N) local i,j,w,t;
  i:=0;
  while i<N do
    for j from 0 to 2^r-1 do
      w:=T[j];
      for t from 0 to s-1 do
        ibutterfly(A,i+t,i+s+t,w);
      od;
      i:=i+2*s;
    od;
  od; end;
```

$$ifftround := \mathbf{proc}(A, T, r, s, N) \tag{9.2.48.2}$$

```
  local i, j, w, t;
  i := 0;
  while i < N do
    for j from 0 to 2^r - 1 do
      w := T[j];
      for t from 0 to s - 1 do
        ibutterfly(A, i+t, i+s+t, w)
      end do;
      i := i+2*s
    end do
  end do
end proc
```

```
> #
  # This is a several-dimensional fft procedure. It works
  # on the array A having size 2^(k[1]+k[2]+..+k[n]),
  # i.e. as if it would have dimensions k[1],..k[n].
  #
```

```
ffts:=proc(A,T,k) local i,r,s,N;
N:=convert(k,`+`); N:=2^N; s:=N;
for i to nops(k) do
  for r from 0 to k[i]-1 do
    s:=s/2;
    fftround(A,T,r,s,N);
  od;
od; end;
```

$$ffts := \mathbf{proc}(A, T, k) \tag{9.2.48.3}$$
$$\quad \mathbf{local}\ i, r, s, N;$$
$$\quad N := convert(k, +);$$
$$\quad N := 2\hat{}N;$$
$$\quad s := N;$$
$$\quad \mathbf{for}\ i\ \mathbf{to}\ nops(k)\ \mathbf{do}$$
$$\quad\quad \mathbf{for}\ r\ \mathbf{from}\ 0\ \mathbf{to}\ k[i] - 1\ \mathbf{do}$$
$$\quad\quad\quad s := 1 / 2 * s;$$
$$\quad\quad\quad fftround(A, T, r, s, N)$$
$$\quad\quad \mathbf{end\ do}$$
$$\quad \mathbf{end\ do}$$
$$\mathbf{end\ proc}$$

```
> #
  # This is a several-dimensional ifft procedure. It works
  # on the array A having size 2^(k[1]+k[2]+..+k[n]),
  # i.e. as if it would have dimensions k[1],..k[n].
  #

  iffts:=proc(A,T,k) local i,r,s,N;
  N:=convert(k,`+`); N:=2^N; s:=1;
  for i from nops(k) to 1 by -1 do
    for r from k[i]-1 to 0 by -1 do
      ifftround(A,T,r,s,N);
      s:=s*2;
    od;
  od; end;
```

$$iffts := \mathbf{proc}(A, T, k) \tag{9.2.48.4}$$
$$\quad \mathbf{local}\ i, r, s, N;$$
$$\quad N := convert(k, +);$$
$$\quad N := 2\hat{}N;$$
$$\quad s := 1;$$
$$\quad \mathbf{for}\ i\ \mathbf{from}\ nops(k)\ \mathbf{by}\ -1\ \mathbf{to}\ 1\ \mathbf{do}$$
$$\quad\quad \mathbf{for}\ r\ \mathbf{from}\ k[i] - 1\ \mathbf{by}\ -1\ \mathbf{to}\ 0\ \mathbf{do}$$
$$\quad\quad\quad ifftround(A, T, r, s, N);$$

```
            s := 2 * s
        end do
    end do
end proc
```

```
> A:=array(0..2^k-1); for i from 0 to 2^k -1 do A[i]:=0 od: A
  [1]:=1.+I;
  print(A);
```

$$A := array(0..31, [\,])$$

$$A_1 := 1. + 1.\,I$$

$$ARRAY([0..31], [0 = 0,\ 1 = 1. + 1.\,I,\ 2 = 0,\ 3 = 0,\ 4 = 0,\ 5 = 0,\ 6 = 0,\ 7 = 0,\ 8 = 0,\ 9 = 0,\ 10 = 0,\ 11 = 0,\ 12 = 0,\ 13 = 0,\ 14 = 0,\ 15 = 0,\ 16 = 0,\ 17 = 0,\ 18 = 0,\ 19 = 0,\ 20 = 0,\ 21 = 0,\ 22 = 0,\ 23 = 0,\ 24 = 0,\ 25 = 0,\ 26 = 0,\ 27 = 0,\ 28 = 0,\ 29 = 0,\ 30 = 0,\ 31 = 0]) \tag{9.2.48.5}$$

```
> ffts(A,T,[2,3]): print(A);
```

$$ARRAY([0..31], [0 = 1. + 1.\,I,\ 1 = -1. - 1.\,I,\ 2 = 1. - 1.\,I,\ 3 = -1. + 1.\,I,\ 4 = 1.414213562 + 0.\,I,\ 5 = -1.414213562 - 0.\,I,\ 6 = 0. - 1.414213562\,I,\ 7 = 0. + 1.414213562\,I,\ 8 = 1. + 1.\,I,\ 9 = -1. - 1.\,I,\ 10 = 1. - 1.\,I,\ 11 = -1. + 1.\,I,\ 12 = 1.414213562 + 0.\,I,\ 13 = -1.414213562 - 0.\,I,\ 14 = 0. - 1.414213562\,I,\ 15 = 0. + 1.414213562\,I,\ 16 = 1. + 1.\,I,\ 17 = -1. - 1.\,I,\ 18 = 1. - 1.\,I,\ 19 = -1. + 1.\,I,\ 20 = 1.414213562 + 0.\,I,\ 21 = -1.414213562 + 0.\,I,\ 22 = 0. - 1.414213562\,I,\ 23 = 0. + 1.414213562\,I,\ 24 = 1. + 1.\,I,\ 25 = -1. - 1.\,I,\ 26 = 1. - 1.\,I,\ 27 = -1. + 1.\,I,\ 28 = 1.414213562 + 0.\,I,\ 29 = -1.414213562 + 0.\,I,\ 30 = 0. - 1.414213562\,I,\ 31 = 0. + 1.414213562\,I]) \tag{9.2.48.6}$$

```
> iffts(A,T,[2,3]): print(A);
```

$$ARRAY([0..31], [0 = 0. + 0.\,I,\ 1 = 32.00000000 + 32.00000000\,I,\ 2 = 0. + 0.\,I,\ 3 = 0. + 0.\,I,\ 4 = 0. + 0.\,I,\ 5 = 8.\,10^{-9} + 8.\,10^{-9}\,I,\ 6 = 0. + 0.\,I,\ 7 = 0. + 0.\,I,\ 8 = 0. + 0.\,I,\ 9 = 0. + 0.\,I,\ 10 = 0. + 0.\,I,\ 11 = 0. + 0.\,I,\ 12 = 0. + 0.\,I,\ 13 = 0. + 0.\,I,\ 14 = 0. + 0.\,I,\ 15 = 0. + 0.\,I,\ 16 = 0. + 0.\,I,\ 17 = 0. + 0.\,I,\ 18 = 0. + 0.\,I,\ 19 = 0. + 0.\,I,\ 20 = 0. + 0.\,I,\ 21 = 0. + 0.\,I,\ 22 = 0. + 0.\,I,\ 23 = 0. + 0.\,I,\ 24 = 0. + 0.\,I,\ 25 = 0. + 0.\,I,\ 26 = 0. + 0.\,I,\ 27 = 0. + 0.\,I,\ 28 = 0. + 0.\,I,\ 29 = 0. + 0.\,I,\ 30 = 0. + 0.\,I,\ 31 = 0. + 0.\,I]) \tag{9.2.48.7}$$

$+0.\mathrm{I}, 30 = 0. + 0.\mathrm{I}, 31 = 0. + 0.\mathrm{I}]\big)$

▼ *9.2.49. Hangtömörítés.

▼ *9.2.50. Képtömörítés.

▼ *9.2.51. PNG.

▼ *9.2.52. JPEG

▼ *9.2.53. DjVu.

▼ *9.2.54. Mozgókép–tömörítés: MPEG.

▶ 9.2.55. További feladatok megoldásokkal.

▼ 9.2.56. További feladatok.

▼ 9.3. Hibakorlátozó kódolás

```
> restart; with(StringTools);
```

[*Anagrams*, *AndMap*, *ApproximateSearch*, *ApproximateSearchAll*, *ArithmeticMean*, *Border*, *BorderArray*, *BorderLength*, *CamelCase*, *Capitalize*, *CaseJoin*, *CaseSplit*, *Center*, *Centre*, *Char*, *CharacterFrequencies*, *CharacterMap*, *Chomp*, *Chop*, *CommonPrefix*, *CommonSuffix*, *Compare*, *CompareCI*, *CountCharacterOccurrences*, *Decode*, *Delete*, *Drop*, *EditDistance*, *Encode*, *Entropy*, *Escape*, *Exchange*, *ExpandCharacterClass*, *ExpandTabs*, *Explode*, *Fence*, *Fibonacci*, *Fill*, *FirstFromLeft*, *FirstFromRight*, *FormatMessage*, *FormatTime*, *FromByteArray*, *Generate*, *Group*, *HammingDistance*, *HammingSearch*, *HammingSearchAll*, *HasASCII*, *HasAlpha*, *HasAlphaNumeric*, *HasBinaryDigit*, *HasControlCharacter*, *HasDigit*, *HasGraphic*, *HasHexDigit*, *HasIdentifier*, *HasIdentifier1*, *HasLower*, *HasOctalDigit*, *HasPrintable*, *HasPunctuation*, *HasSpace*, *HasUpper*, *HasVowel*, *Hash*, *Implode*, *Insert*, I, *IsASCII*, *IsAlpha*, *IsAlphaNumeric*, *IsAnagram*, *IsBalanced*, *IsBinaryDigit*, *IsConjugate*, *IsControlCharacter*, *IsDigit*, *IsGraphic*, *IsHexDigit*, *IsIdentifier*, *IsIdentifier1*, *IsLower*, *IsMonotonic*, *IsOctalDigit*, *IsPalindrome*, *IsPeriod*, *IsPermutation*, *IsPrefix*, *IsPrimitive*, *IsPrintable*, *IsPunctuation*, *IsSorted*, *IsSpace*, *IsSubSequence*, *IsSuffix*, *IsUpper*, *IsVowel*, *Join*, *LeftFold*, *LeftRecursivePathOrder*, *Length*, *Levenshtein*, *LexOrder*, *LongestCommonSubSequence*, *LongestCommonSubString*, *LowerCase*, *LyndonFactors*, *Map*, *MaxChar*, *MaximalPalindromicSubstring*, *Metaphone*, *MinChar*, *MinimumConjugate*, *MonotonicFactors*, *NGrams*, (9.3.1)

NthWord, *OrMap*, *Ord*, *Overlap*, *PadLeft*, *PadRight*, *ParseTime*, *PatternDictionary*, *Period*, *Permute*, *PrefixDistance*, *PrimitiveRoot*, *Random*, *Randomize*, *RegMatch*, *RegSplit*, *RegSub*, *RegSubs*, *Remove*, *Repeat*, *RevLexOrder*, *Reverse*, *RightFold*, *RightRecursivePathOrder*, *Rotate*, *Search*, *SearchAll*, *Select*, *SelectRemove*, *Shift*, *ShortLexOrder*, *ShortRevLexOrder*, *SimilarityCoefficient*, *Sort*, *Soundex*, *Split*, *Squeeze*, *Stem*, *StringBuffer*, *SubString*, *Substitute*, *SubstituteAll*, *SuffixDistance*, *Support*, *SyllableLength*, *Tabulate*, *Take*, *ThueMorse*, *ToByteArray*, *Trim*, *TrimLeft*, *TrimRight*, *UpperCase*, *Visible*, *WildcardMatch*, *WordCount*, *Words*, *WrapText*]

9.3.1. Paritásbites kód.

```
> addparity:=proc(m::string) local i,c; c:=0;
  for i to length(m) do
    if m[i]="1" then c:=1-c; next; fi;
    if m[i]<>"0" then return FAIL fi;
  od; if c=1 then cat(m,"1") else cat(m,"0") fi; end;
```

```
addparity := proc(m::string)                                    (9.3.1.1)
    local i, c;
    c := 0;
    for i to length(m) do
        if m[i] = "1" then
            c := 1 - c;
            next
        end if;
        if m[i] <> "0" then
            return FAIL
        end if
    end do;
    if c = 1 then
        cat(m, "1")
    else
        cat(m, "0")
    end if
end proc
```

```
> addparity("0110"); addparity("0010");
```

```
                              "01100"
                              "00101"                           (9.3.1.2)
```

9.3.2. Példa.

9.3.3. Példa: paritásbites kód.

9.3.4. Hibajelző kód.

9.3.5. Kódok távolsága és súlya.

```
> HammingDistance("abcd","abdd");
```

$$1 \tag{9.3.5.1}$$

9.3.6. Minimális távolságú dekódolás.

```
> Dt:=table(): Dt["0000"]:=FAIL: Dt["0001"]:=FAIL: Dt["0010"]
  :="0010":
  Dt["0011"]:="0010": Dt["0100"]:="0100": Dt["0101"]:="0100":
  Dt["0110"]:=FAIL: Dt["0111"]:="1111": Dt["1000"]:=FAIL:
  Dt["1001"]:="1111": Dt["1010"]:="0010": Dt["1011"]:="1111":
  Dt["1100"]:="0100": Dt["1101"]:="1111": Dt["1110"]:="1111":
  Dt["1111"]:="1111":

  Dt["0011"]; Dt["0000"]; Dt["1110"];
```

$$\text{"0010"}$$

$$\mathit{FAIL}$$

$$\text{"1111"} \tag{9.3.6.1}$$

9.3.7. Hibajavító kód.

*9.3.8. Hibajavítás ismert hibahelyekkel.

9.3.9. Ismétléses kód.

9.3.10. Kétdimenziós paritásellenőrzés.

```
> m:=9: n:=8: rnd:=rand(0..1): A:=Matrix(m,n):
  for i to m do for j to n do A[i,j]:=rnd(): od: od: A;
```

(9.3.10.1)

$$\begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 \end{bmatrix} \tag{9.3.10.1}$$

```
> B:=Matrix(m+1,n+1):
  for i to m do s:=0: for j to n do B[i,j]:=A[i,j]: if A[i,j]
  <>0 then s:=1-s fi: od: B[i,n+1]:=s od:
  for j to n+1 do s:=0: for i to m do if B[i,j]<>0 then s:=1-
  s; fi: od: B[m+1,j]:=s: od:
  B;
```

$$\begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 1 \end{bmatrix} \tag{9.3.10.2}$$

▼ *9.3.11. Hamming-korlát.*

▼ *9.3.12. Singleton-korlát.*

▶ *9.3.13. Lineáris kód.*

▶ *9.3.14. Generátormátrix, ellenőrző mátrix.*

▶ *9.3.15. Szindrómadekódolás.*

▼ *9.3.16. Példa: Fano-kód.*

```
> isvectorspacemod2:=proc(S) local x,y,z;
  for x in S do for y in S do z:=zip((xx,yy)->xx+yy mod 2,x,
  y);
    if not z in S then return false fi; od; od; true; end;
```

```
isvectorspacemod2 := proc(S)                                    (9.3.16.1)
    local x, y, z;
    for x in S do
        for y in S do
            z := zip(proc(xx, yy)
                option operator, arrow;
                mod(xx + yy, 2)
            end proc, x, y);
            if not in(z, S) then
                return false
            end if
        end do
    end do;
    true
end proc
```

```
> Fano:={[0,0,0,0,0,0,0],[1,1,1,1,1,1,1],[1,1,1,0,0,0,0],[0,
  0,0,1,1,1,1],[1,0,0,1,1,0,0],[0,1,1,0,0,1,1],[0,1,0,1,0,1,
  0],[1,0,1,0,1,0,1],[0,0,1,1,0,0,1],[1,1,0,0,1,1,0],[1,0,0,
  0,0,1,1],[0,1,1,1,1,0,0],[0,1,0,0,1,0,1],[1,0,1,1,0,1,0],
  [0,0,1,0,1,1,0],[1,1,0,1,0,0,1]};

  isvectorspacemod2(Fano);
```

```
Fano := {[0, 0, 0, 0, 0, 0, 0], [1, 1, 1, 1, 1, 1, 1], [1, 1, 1, 0, 0, 0,
    0], [0, 0, 0, 1, 1, 1, 1], [1, 0, 0, 1, 1, 0, 0], [0, 1, 1, 0, 0, 1, 1], [0,
    1, 0, 1, 0, 1, 0], [1, 0, 1, 0, 1, 0, 1], [0, 0, 1, 1, 0, 0, 1], [1, 1, 0, 0,
    1, 1, 0], [1, 0, 0, 0, 0, 1, 1], [0, 1, 1, 1, 1, 0, 0], [0, 1, 0, 0, 1, 0,
    1], [1, 0, 1, 1, 0, 1, 0], [0, 0, 1, 0, 1, 1, 0], [1, 1, 0, 1, 0, 0, 1]}
                                true                                (9.3.16.2)
```

```
> with(linalg);
```

```
[BlockDiagonal, GramSchmidt, JordanBlock, LUdecomp, QRdecomp,      (9.3.16.3)
    Wronskian, addcol, addrow, adj, adjoint, angle, augment,
    backsub, band, basis, bezout, blockmatrix, charmat, charpoly,
    cholesky, col, coldim, colspace, colspan, companion, concat, cond,
    copyinto, crossprod, curl, definite, delcols, delrows, det, diag,
    diverge, dotprod, eigenvals, eigenvalues, eigenvectors,
    eigenvects, entermatrix, equal, exponential, extend, ffgausselim,
```

fibonacci, forwardsub, frobenius, gausselim, gaussjord, geneqns, genmatrix, grad, hadamard, hermite, hessian, hilbert, htranspose, ihermite, indexfunc, innerprod, intbasis, inverse, ismith, issimilar, iszero, jacobian, jordan, kernel, laplacian, leastsqrs, linsolve, matadd, matrix, minor, minpoly, mulcol, mulrow, multiply, norm, normalize, nullspace, orthog, permanent, pivot, potential, randmatrix, randvector, rank, ratform, row, rowdim, rowspace, rowspan, rref, scalarmul, singularvals, smith, stackmatrix, submatrix, subvector, sumbasis, swapcol, swaprow, sylvester, toeplitz, trace, transpose, vandermonde, vecpotent, vectdim, vector, wronskian]

```
> rangemod2:=proc(A::matrix) local m,n,i,j,v,S;
  m:=rowdim(A); n:=coldim(A); S:={};
  for i from 0 to 2^n-1 do
    v:=convert(i,base,2); v:=[op(v),0$j=nops(v)+1..n]; v:=
  vector(n,v);
    v:=multiply(A,v); v:=map(x->x mod 2,v);
    v:=convert(v,list); S:=S union {v};
  od; S; end;
```

```
rangemod2 := proc(A::(linalg:-matrix))                    (9.3.16.4)
  local m, n, i, j, v, S;
  m := linalg:-rowdim(A);
  n := linalg:-coldim(A);
  S := {};
  for i from 0 to 2^n - 1 do
    v := convert(i, base, 2);
    v := [op(v), $(0, j = nops(v) + 1..n)];
    v := linalg:-vector(n, v);
    v := linalg:-multiply(A, v);
    v := map(proc(x)
        option operator, arrow;
        mod(x, 2)
      end proc, v);
    v := convert(v, list);
    S := union(S, {v})
  end do;
  S
end proc
```

```
> G:=matrix([[1,1,1,1],[1,0,0,1],[1,0,1,1],[0,0,1,1],[0,0,0,
```

```
1],[0,1,1,1],[0,1,0,1]]);
```

$$G := \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \\ 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 \end{bmatrix} \tag{9.3.16.5}$$

```
> rangemod2(G); evalb(Fano=%);
```

$\{[0, 0, 0, 0, 0, 0, 0], [1, 1, 1, 1, 1, 1, 1], [1, 1, 1, 0, 0, 0, 0], [0, 0, 0, 1, 1, 1, 1], [1, 0, 0, 1, 1, 0, 0], [0, 1, 1, 0, 0, 1, 1], [0, 1, 0, 1, 0, 1, 0], [1, 0, 1, 0, 1, 0, 1], [0, 0, 1, 1, 0, 0, 1], [1, 1, 0, 0, 1, 1, 0], [1, 0, 0, 0, 0, 1, 1], [0, 1, 1, 1, 1, 0, 0], [0, 1, 0, 0, 1, 0, 1], [1, 0, 1, 1, 0, 1, 0], [0, 0, 1, 0, 1, 1, 0], [1, 1, 0, 1, 0, 0, 1]\}$

$$true \tag{9.3.16.6}$$

```
> addcol(G,1,2,1):addcol(%,1,3,1):addcol(%,1,4,1):map(x->x
  mod 2,%);
```

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 \\ 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \\ 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 \end{bmatrix} \tag{9.3.16.7}$$

```
> addcol(%,2,1,1):addcol(%,2,3,1):map(x->x mod 2,%);
```

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \\ 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 1 \end{bmatrix} \tag{9.3.16.8}$$

```
> addcol(%,3,2,1):map(x->x mod 2,%);
```

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 \\ 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 1 \end{bmatrix} \tag{9.3.16.9}$$

```
> addcol(%,4,2,1):addcol(%,4,3,1):G1:=map(x->x mod 2,%);
```

$$G1 := \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 \end{bmatrix} \tag{9.3.16.10}$$

```
> rangemod2(G1); evalb(Fano=%);
```

$$\{[0,0,0,0,0,0,0],[1,1,1,1,1,1,1],[1,1,1,0,0,0,0],[0,0,0,1,1,1,1],[1,0,0,1,1,0,0],[0,1,1,0,0,1,1],[0,1,0,1,0,1,0],[1,0,1,0,1,0,1],[0,0,1,1,0,0,1],[1,1,0,0,1,1,0],[1,0,0,0,0,1,1],[0,1,1,1,1,0,0],[0,1,0,0,1,0,1],[1,0,1,1,0,1,0],[0,0,1,0,1,1,0],[1,1,0,1,0,0,1]\}$$

$$true \tag{9.3.16.11}$$

```
> H:=matrix([[0,1,1,1,1,0,0],[1,0,1,1,0,1,0],[1,1,0,1,0,0,1]]
  );
```

$$H := \begin{bmatrix} 0 & 1 & 1 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 1 \end{bmatrix} \tag{9.3.16.12}$$

```
> sindromemod2:=proc(H::matrix,St::table) local i,j,k,m,n,s,
  ss,sss,x;
  m:=rowdim(H); n:=coldim(H); if m>=n then return FAIL fi;
  for i from 0 to 2^m-1 do
    s:=convert(i,base,2); s:=[op(s),0$j=nops(s)+1..m];
    St[s]:=[1$j=1..n]
  od;
```

```
    for i from 0 to 2^n-1 do
      ss:=convert(i,base,2); ss:=[op(ss),0$j=nops(%)+1..n];
      sss:=convert(ss,vector); k:=multiply(H,sss);
      k:=map(x->x mod 2,k); k:=convert(k,list); s:=St[%];
      if convert(ss,`+`)<convert(s,`+`) then St[k]:=ss fi;
    od; end;
```

```
sindromemod2 := proc(H::(linalg:-matrix), St::table)                    (9.3.16.13)
    local i, j, k, m, n, s, ss, sss, x;
    m := linalg:-rowdim(H);
    n := linalg:-coldim(H);
    if n <= m then
        return FAIL
    end if;
    for i from 0 to 2^m - 1 do
        s := convert(i, base, 2);
        s := [op(s), $(0, j = nops(s) + 1 ..m)];
        St[s] := [$(1, j = 1 ..n)]
    end do;
    for i from 0 to 2^n - 1 do
        ss := convert(i, base, 2);
        ss := [op(ss), $(0, j = nops(%) + 1 ..n)];
        sss := convert(ss, linalg:-vector);
        k := linalg:-multiply(H, sss);
        k := map(proc(x)
            option operator, arrow;
            mod(x, 2)
        end proc, k);
        k := convert(k, list);
        s := St[%];
        if convert(ss, +) < convert(s, +) then
            St[k] := ss
        end if
    end do
end proc
```

```
> St:=table(): sindromemod2(H,St); print(St);
```

```
table([([0, 1, 1]) = ([1, 0, 0, 0, 0, 0, 0]), ([0, 0, 1]) = ([0, 0, 0, 0,    (9.3.16.14)
    0, 0, 1]), ([1, 1, 0]) = ([0, 0, 1, 0, 0, 0, 0]), ([0, 1, 0]) = ([0, 0,
    0, 0, 0, 1, 0]), ([0, 0, 0]) = ([0, 0, 0, 0, 0, 0, 0]), ([1, 1,
    1]) = ([0, 0, 0, 1, 0, 0, 0]), ([1, 0, 1]) = ([0, 1, 0, 0, 0, 0, 0]),
```

$$([1, 0, 0]) = ([0, 0, 0, 0, 1, 0, 0])])$$

```
> c:=vector([0,1,0,0,1,0,1]); multiply(H,c); map(x->x mod 2,
  %);
```

$$c := [0 \quad 1 \quad 0 \quad 0 \quad 1 \quad 0 \quad 1]$$
$$[2 \quad 0 \quad 2]$$
$$[0 \quad 0 \quad 0] \tag{9.3.16.15}$$

```
> e:=vector([0,0,0,0,1,0,0]); v:=evalm(c+e); s:=multiply(H,v)
  ;
  s:=convert(s,list) mod 2; St[s];
```

$$e := [0 \quad 0 \quad 0 \quad 0 \quad 1 \quad 0 \quad 0]$$
$$v := [0 \quad 1 \quad 0 \quad 0 \quad 2 \quad 0 \quad 1]$$
$$s := [3 \quad 0 \quad 2]$$
$$s := [1, 0, 0]$$
$$[0, 0, 0, 0, 1, 0, 0] \tag{9.3.16.16}$$

```
> e:=vector([0,0,0,0,1,0,1]); v:=evalm(c+e); s:=multiply(H,v)
  ;
  s:=convert(s,list) mod 2; St[s];
```

$$e := [0 \quad 0 \quad 0 \quad 0 \quad 1 \quad 0 \quad 1]$$
$$v := [0 \quad 1 \quad 0 \quad 0 \quad 2 \quad 0 \quad 2]$$
$$s := [3 \quad 0 \quad 3]$$
$$s := [1, 0, 1]$$
$$[0, 1, 0, 0, 0, 0, 0] \tag{9.3.16.17}$$

```
> e:=vector([1,1,1,0,0,0,0]); v:=evalm(c+e); s:=multiply(H,v)
  ;
  s:=convert(s,list) mod 2; St[s];
```

$$e := [1 \quad 1 \quad 1 \quad 0 \quad 0 \quad 0 \quad 0]$$
$$v := [1 \quad 2 \quad 1 \quad 0 \quad 1 \quad 0 \quad 1]$$
$$s := [4 \quad 2 \quad 4]$$
$$s := [0, 0, 0]$$
$$[0, 0, 0, 0, 0, 0, 0] \tag{9.3.16.18}$$

9.3.17. Példa: Reed–Müller-kódok.

```
> RM51basis:={
  [1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,
  1,1,1],
  [1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,
  0,0,0],
  [1,1,1,1,1,1,1,1,0,0,0,0,0,0,0,0,1,1,1,1,1,1,1,1,0,0,0,0,
  0,0,0],
  [1,1,1,1,0,0,0,0,1,1,1,1,0,0,0,0,1,1,1,1,0,0,0,0,1,1,1,1,0,
  0,0,0],
```

```
   [1,1,0,0,1,1,0,0,1,1,0,0,1,1,0,0,1,1,0,0,1,1,0,0,1,1,0,0,1,
   1,0,0],
   [1,0,1,0,1,0,1,0,1,0,1,0,1,0,1,0,1,0,1,0,1,0,1,0,1,0,1,0,1,
   0,1,0]};
```

$$RM51basis := \{[1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1], [1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0], [1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0], [1, 1, 1, 1, 0, 0, 0, 0, 1, 1, 1, 1, 0, 0, 0, 0, 1, 1, 1, 1, 0, 0, 0, 0, 1, 1, 1, 1, 0, 0, 0, 0], [1, 1, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0], [1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0]\} \quad (9.3.17.1)$$

```
> linenvelop:=proc(S) local x,y,SS,SSS; SS:=S;
  while true do
    SSS:=SS;
    for x in SS do for y in SS do
      SSS:=SSS union {zip((xx,yy)->xx+yy mod 2,x,y)}
    od; od;
    if SS=SSS then return(SS) fi;
    SS:=SSS;
  od; end;
```

$$linenvelop := \mathbf{proc}(S) \quad (9.3.17.2)$$

```
local x, y, SS, SSS;
SS := S;
do
    SSS := SS;
    for x in SS do
        for y in SS do
            SSS := union(SSS, {zip(proc(xx, yy)
                option operator, arrow;
                mod(xx + yy, 2)
            end proc, x, y)})
        end do
    end do;
    if SS = SSS then
        return SS
    end if;
    SS := SSS
end do
end proc
```

```
> linenvelop(RM51basis); nops(%);
```

{[1, 1, 0, 0, 0, 0, 1, 1, 1, 1, 0, 0, 0, 0, 1, 1, 1, 1, 0, 0, 0, 0, 1, 1, 1, 1, 0, 0, 0, 0, 1, 1], [1, 0, 1, 0, 0, 1, 0, 1, 1, 0, 1, 0, 0, 1, 0, 1, 1, 0, 1, 0, 0, 1, 0, 1, 1, 0, 1, 0, 0, 1, 0, 1], [1, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 1], [1, 1, 1, 1, 0, 0, 0, 0, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 0, 0, 0, 0, 1, 1, 1, 1], [1, 1, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 1, 1], [1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1], [1, 0, 0, 1, 0, 1, 1, 0, 0, 1, 1, 0, 1, 0, 0, 1, 0, 1, 1, 0, 0, 1, 1, 0, 1, 0, 0, 1, 1, 0, 0, 1], [1, 0, 0, 1, 1, 0, 0, 1, 0, 1, 1, 0, 0, 1, 1, 0, 1, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 1], [1, 0, 0, 1, 0, 1, 1, 0, 0, 1, 1, 0, 1, 0, 0, 1, 1, 0, 0, 1, 0, 1, 1, 0, 0, 1, 1, 0, 1, 0, 0, 1], [0, 1, 0, 1, 0, 1, 0, 1, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0], [0, 0, 1, 1, 1, 1, 0, 0, 0, 0, 1, 1, 1, 1, 0, 0, 1, 1, 0, 0, 0, 0, 1, 1, 1, 1, 0, 0, 0, 0, 1, 1], [0, 1, 0, 1, 1, 0, 1, 0, 0, 1, 0, 1, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 0, 1, 0, 1, 0, 1, 0, 1], [0, 1, 1, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 1, 1, 0], [0, 1, 1, 0, 1, 0, 0, 1, 0, 1, 1, 0, 1, 0, 0, 1, 0, 1, 1, 0, 1, 0, 0, 1, 0, 1, 1, 0, 0, 1, 0, 1], [0, 1, 1, 0, 1, 0, 0, 1, 1, 0, 0, 1, 0, 1, 1, 0, 1, 0, 0, 1, 1, 0, 0, 1, 0, 1, 1, 0, 1, 0, 0, 1], [0, 0, 1, 1, 1, 1, 0, 0, 1, 1, 0, 0, 0, 0, 1, 1, 1, 1, 0, 0, 1, 1, 0, 0, 0, 0, 1, 1, 1, 1, 0, 0], [0, 1, 0, 1, 1, 0, 1, 0, 1, 0, 1, 0, 0, 1, 0, 1, 0, 1, 0, 1, 1, 0, 1, 0, 0, 1, 0, 1, 0, 1, 0, 1], [0, 1, 0, 1, 1, 0, 1, 0, 0, 1, 0, 1, 0, 1, 0, 1, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0], [1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1], [0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0], [0, 0, 1, 1, 0, 0, 1, 1, 1, 1, 0, 0, 1, 1, 0, 0, 0, 0, 1, 1, 0, 0, 1, 1, 1, 1, 0, 0, 1, 1, 0, 0], [0, 1, 0, 1, 0, 1, 0, 1, 1, 0, 1, 0, 1, 0, 1, 0, 0, 1, 0, 1, 0, 1, 0, 1, 1, 0, 1, 0, 1, 0, 1, 0], [0, 0, 1, 1, 1, 0, 0, 1, 1, 0, 0, 0, 0, 1, 1, 0, 0, 1, 1, 1, 1, 0, 0, 1, 1, 0, 0, 0, 0, 1, 1, 1], [0, 1, 0, 1, 1, 0, 1, 0, 1, 0, 1, 0, 0, 1, 0, 1, 0, 1, 0, 1, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0], [1, 1, 0, 0, 1, 1, 0, 0, 0, 0, 1, 1, 0, 0, 1, 1, 1, 1, 0, 0, 1, 1, 0, 0, 0, 0, 1, 1, 0, 0, 1, 1], [1, 0, 0, 1, 0, 1, 1, 0, 1, 0, 0, 1, 0, 1, 1, 0, 1, 0, 0, 1, 1, 0, 0, 1, 0, 1, 1, 0, 1, 0, 0, 1], [1, 0, 1, 0, 0, 1, 0, 1, 0, 1, 1, 0, 1, 0, 0, 1, 0, 1, 1, 0, 1, 0, 1, 0, 0, 1, 0, 1, 0, 1, 0, 1], [1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1], [1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0], [1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0], [1, 1, 1, 1, 0, 0, 0, 0, 1, 1, 1, 1, 0, 0, 0, 0, 1, 1, 1, 1, 0, 0, 0, 0, 1, 1, 1, 1, 0, 0, 0, 0], [1, 1, 0,

0, 1, 1, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0], [1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0], [1, 1, 0, 0, 1, 1, 0, 0, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 1, 1, 1, 1, 0, 0, 1, 1, 0, 0], [1, 0, 0, 1, 1, 0, 0, 1, 0, 1, 1, 0, 0, 1, 1, 0, 0, 1, 0, 1, 1, 0, 0, 1, 1, 0], [0, 0, 1, 1, 0, 0, 1, 1, 1, 1, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 0, 0, 1, 1, 0, 0, 1, 1], [1, 0, 0, 1, 0, 1, 1, 0, 1, 0, 0, 1, 0, 1, 1, 0, 1, 0, 0, 1, 0, 1, 1, 0, 1, 0, 0, 1, 0, 1, 1, 0], [1, 1, 0, 0, 0, 0, 1, 1, 1, 1, 0, 0, 0, 0, 1, 1, 0, 0, 1, 1, 1, 1, 0, 0, 0, 0, 1, 1, 1, 1, 0, 0], [1, 0, 1, 0, 0, 1, 0, 1, 1, 0, 1, 0, 0, 1, 0, 1, 0, 1, 0, 1, 1, 0, 1, 0, 0, 1, 0, 1, 1, 0, 1, 0], [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0], [0, 1, 1, 0, 1, 0, 0, 1, 0, 1, 1, 0, 1, 0, 0, 1, 1, 0, 0, 1, 0, 1, 1, 0, 1, 0, 0, 1, 0, 1, 1, 0], [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1], [0, 0, 1, 1, 1, 1, 0, 0, 0, 0, 1, 1, 1, 1, 0, 0, 0, 0, 1, 1, 1, 1, 0, 0, 0, 0, 1, 1, 1, 1, 0, 0], [0, 1, 0, 1, 1, 0, 1, 0, 0, 1, 0, 1, 1, 0, 1, 0, 0, 1, 0, 1, 1, 0, 1, 0, 0, 1, 0, 1, 1, 0, 1, 0], [0, 1, 1, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 1, 1, 0], [0, 0, 0, 0, 1, 1, 1, 1, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 1, 1, 1, 1, 0, 0, 0, 0], [0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 1, 1, 1, 1, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0], [0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1], [0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1], [0, 1, 1, 0, 0, 1, 1, 0, 1, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 1, 0, 1, 1, 0, 0, 1, 1, 0], [0, 0, 0, 0, 1, 1, 1, 1, 0, 0, 0, 0, 1, 1, 1, 1, 0, 0, 0, 0, 1, 1, 1, 1, 0, 0, 0, 0, 1, 1, 1, 1], [0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 1, 1, 1, 1, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0], [0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1], [0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1], [0, 1, 1, 0, 0, 1, 1, 0, 1, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 1, 0, 1, 1, 0, 0, 1, 1, 0], [0, 0, 0, 0, 1, 1, 1, 1, 0, 0, 0, 0, 1, 1, 1, 1, 0, 0, 0, 0, 1, 1, 1, 1, 0, 0, 0, 0, 1, 1, 1, 1], [1, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 1], [0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 1, 1], [1, 1, 0, 0, 0, 0, 1, 1, 0, 0, 1, 1, 1, 1, 0, 0, 0, 0, 1, 1, 1, 1, 0, 0, 0, 0, 1, 1, 1, 1, 0, 0], [0, 1, 1, 0, 0, 1, 1, 0, 1, 0, 0, 1, 1, 0, 0, 1, 0, 1, 1, 0, 1, 0, 0, 1, 1, 0, 0, 1, 0, 1, 1, 0], [0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1], [1, 0, 1, 0, 1, 0, 1, 0, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1], [0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0], [1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0], [1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0,

0], [1, 1, 0, 0, 0, 0, 1, 1, 0, 0, 1, 1, 1, 1, 0, 0, 1, 1, 0, 0, 0, 0, 1, 1, 0, 0, 1, 1, 1, 1, 0, 0], [1, 0, 1, 0, 0, 1, 0, 1, 0, 1, 0, 1, 1, 0, 1, 0, 1, 0, 1, 0, 0, 1, 0, 1, 0, 1, 0, 1, 1, 0, 1, 0], [0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1]}

64 (9.3.17.3)

*9.3.18. Hamming-kódok.

*9.3.19. Feladat.

9.3.20. Polinomkódok.

9.3.21. CRC-kódok.

```
> CRC1:=modp1(ConvertIn(x+1,x),2);
```

$$CRC1 := (x+1) \bmod 2 \qquad (9.3.21.1)$$

```
> CRC5USB:=modp1(ConvertIn(x^5+x^2+1,x),2);
```

$$CRC5USB := (x^5+x^2+1) \bmod 2 \qquad (9.3.21.2)$$

```
> m:=5: k:=10: L:=[0,1,1,0,1,1,1,1,1,1]; [0$'i'=1..m,op(L)];
  l:=modp1(ConvertIn(%,x),2); modp1(Rem(%,CRC5USB),2);
  modp1(Add(l,%),2); modp1(ConvertOut(%),2);
```

$$L := [0, 1, 1, 0, 1, 1, 1, 1, 1, 1]$$

$$[0, 0, 0, 0, 0, 0, 1, 1, 0, 1, 1, 1, 1, 1, 1]$$

$$l := (x^{14}+x^{13}+x^{12}+x^{11}+x^{10}+x^9+x^7+x^6) \bmod 2$$

$$(x^4+x^3+x^2+1) \bmod 2$$

$$(x^{14}+x^{13}+x^{12}+x^{11}+x^{10}+x^9+x^7+x^6+x^4+x^3+x^2+1) \bmod 2$$

$$[1, 0, 1, 1, 1, 0, 1, 1, 0, 1, 1, 1, 1, 1, 1] \qquad (9.3.21.3)$$

```
> CRC8:=modp1(ConvertIn(x^8+x^2+x+1,x),2);
```

$$CRC8 := (x^8+x^2+x+1) \bmod 2 \qquad (9.3.21.4)$$

```
> CRC16IBM:=modp1(ConvertIn(x^16+x^15+x^2+1,x),2);
```

$$CRC16IBM := (x^{16}+x^{15}+x^2+1) \bmod 2 \qquad (9.3.21.5)$$

```
> CRC16CCITT:=modp1(ConvertIn(x^16+x^12+x^5+1,x),2);
```

$$CRC16CCITT := (x^{16}+x^{12}+x^5+1) \bmod 2 \qquad (9.3.21.6)$$

```
> CRC32:=modp1(ConvertIn(x^32+x^26+x^23+x^22+x^16+x^12+x^11+
  x^10+x^8+x^7+x^5+x^4+x^2+x+1,x),2);
```

$$CRC32 := (x^{32}+x^{26}+x^{23}+x^{22}+x^{16}+x^{12}+x^{11}+x^{10}+x^8+x^7+x^5+x^4+x^2+x+1) \bmod 2 \qquad (9.3.21.7)$$

```
> CRC64ISO:=modp1(ConvertIn(x^64+x^4+x^3+x+1,x),2);
```

$$CRC64ISO := (x^{64}+x^4+x^3+x+1) \bmod 2 \qquad (9.3.21.8)$$

*9.3.22. A CRC-kódok hibajelző képessége.

```
> modp1(Factors(CRC1),2); m:=1; ifactor(2^m-1);
```

$$[1, [[(x+1) \bmod 2, 1]]]$$

$$m := 1$$

$$1 \tag{9.3.22.1}$$

```
> modp1(Factors(CRC5USB),2); m:=5; ifactor(2^m-1);
```

$$[1, [[(x^5+x^2+1) \bmod 2, 1]]]$$

$$m := 5$$

$$(31) \tag{9.3.22.2}$$

```
> p:=modp1(ConvertIn(x,x),2); modp1(Powmod(p,31,CRC5USB),2);
```

$$p := x \bmod 2$$

$$1 \bmod 2 \tag{9.3.22.3}$$

```
> modp1(Factors(CRC8),2); m:=7; ifactor(2^m-1);
```

$$[1, [[(x+1) \bmod 2, 1], [(x^7+x^6+x^5+x^4+x^3+x^2+1) \bmod 2, 1]]]$$

$$m := 7$$

$$(127) \tag{9.3.22.4}$$

```
> modp1(Factors(CRC16IBM),2); m:=15; ifactor(2^m-1);
```

$$[1, [[(x+1) \bmod 2, 1], [(x^{15}+x+1) \bmod 2, 1]]]$$

$$m := 15$$

$$(7)\,(31)\,(151) \tag{9.3.22.5}$$

```
> modp1(Powmod(p,32767,CRC16IBM),2);
  modp1(Powmod(p,32767/7,CRC16IBM),2);
  modp1(Powmod(p,32767/31,CRC16IBM),2);
  modp1(Powmod(p,32767/151,CRC16IBM),2);
```

$$1 \bmod 2$$

$$(x^{15}+x^{12}+x^{10}+x^9+x^5+x^4+1) \bmod 2$$

$$(x^{15}+x^9+x^6+x^5+x^4+x+1) \bmod 2$$

$$(x^{15}+x^{13}+x^{11}+x^9+x^6+x^5+x^4+x^3+x^2) \bmod 2 \tag{9.3.22.6}$$

```
> modp1(Factors(CRC16CCITT),2); m:=15; ifactor(2^m-1);
```

$$[1, [[(x+1) \bmod 2, 1], [(x^{15}+x^{14}+x^{13}+x^{12}+x^4+x^3+x^2+x+1) \bmod 2, 1]]]$$

$$m := 15$$

$$(7)\,(31)\,(151) \tag{9.3.22.7}$$

```
> modp1(Powmod(p,32767,CRC16CCITT),2);
  modp1(Powmod(p,32767/7,CRC16CCITT),2);
  modp1(Powmod(p,32767/31,CRC16CCITT),2);
  modp1(Powmod(p,32767/151,CRC16CCITT),2);
```

$$1 \bmod 2$$

$$\left(x^{13}+x^{11}+x^{10}+x^6+x\right) \bmod 2$$
$$\left(x^{15}+x^{14}+x^{13}+x^{12}+x^9+x^7+x^5+x^3+x^2+x+1\right) \bmod 2$$
$$\left(x^{14}+x^{11}+x^7+x^6+x^5+x+1\right) \bmod 2 \qquad (9.3.22.8)$$

```
> modp1(Factors(CRC32),2); m:=32; ifactor(2^m-1);
```

$$\left[1, \left[\left[\left(x^{32}+x^{26}+x^{23}+x^{22}+x^{16}+x^{12}+x^{11}+x^{10}+x^8+x^7+x^5+x^4+x^2+x+1\right) \bmod 2, 1\right]\right]\right]$$
$$m := 32$$
$$(3)\,(5)\,(17)\,(257)\,(65537) \qquad (9.3.22.9)$$

```
> modp1(Powmod(p,2^32-1,CRC32),2);
  modp1(Powmod(p,(2^32-1)/3,CRC32),2);
  modp1(Powmod(p,(2^32-1)/5,CRC32),2);
  modp1(Powmod(p,(2^32-1)/17,CRC32),2);
  modp1(Powmod(p,(2^32-1)/257,CRC32),2);
  modp1(Powmod(p,(2^32-1)/65537,CRC32),2);
```

$$1 \bmod 2$$
$$\left(x^{30}+x^{27}+x^{25}+x^{23}+x^{22}+x^{20}+x^{18}+x^{17}+x^{16}+x^{14}+x^{13}+x^{11}+x^{10}+x^7+x^6+x^5+x+1\right) \bmod 2$$
$$\left(x^{28}+x^{27}+x^{26}+x^{25}+x^{22}+x^{19}+x^{15}+x^5+1\right) \bmod 2$$
$$\left(x^{26}+x^{24}+x^{21}+x^{20}+x^{19}+x^{18}+x^{12}+x^{10}+x^5+x^4+1\right) \bmod 2$$
$$\left(x^{31}+x^{28}+x^{25}+x^{24}+x^{22}+x^{18}+x^{13}+x^{12}+x^{11}+x^7+x^5+x^3+x^2\right) \bmod 2$$
$$\left(x^{31}+x^{30}+x^{29}+x^{28}+x^{27}+x^{25}+x^{21}+x^{20}+x^{19}+x^{17}+x^{15}+x^{14}+x^{12}+x^{11}+x^{10}+x^9+x^5+x^4+x^3+x+1\right) \bmod 2 \qquad (9.3.22.10)$$

```
> modp1(Factors(CRC64ISO),2); m:=64; ifactor(2^m-1);
```

$$\left[1, \left[\left[\left(x^{64}+x^4+x^3+x+1\right) \bmod 2, 1\right]\right]\right]$$
$$m := 64$$
$$(3)\,(5)\,(17)\,(257)\,(641)\,(6700417)\,(65537) \qquad (9.3.22.11)$$

```
> modp1(Powmod(p,2^64-1,CRC64ISO),2);
  modp1(Powmod(p,(2^64-1)/3,CRC64ISO),2);
  modp1(Powmod(p,(2^64-1)/5,CRC64ISO),2);
  modp1(Powmod(p,(2^64-1)/17,CRC64ISO),2);
  modp1(Powmod(p,(2^64-1)/257,CRC64ISO),2);
  modp1(Powmod(p,(2^64-1)/65537,CRC64ISO),2);
  modp1(Powmod(p,(2^64-1)/6700417,CRC64ISO),2);
```

$$1 \bmod 2$$
$$\left(x^{60}+x^{59}+x^{56}+x^{55}+x^{54}+x^{51}+x^{48}+x^{45}+x^{44}+x^{42}+x^{41}+x^{39}+x^{36}+x^{35}+x^{34}+x^{33}+x^{32}+x^{29}+x^{26}+x^{25}+x^{24}+x^{23}+x^{19}+x^{17}+x^{15}+x^{14}+x^{12}+x^{11}+x^{10}+x\right) \bmod 2$$
$$\left(x^{62}+x^{60}+x^{59}+x^{58}+x^{56}+x^{55}+x^{53}+x^{52}+x^{51}+x^{46}+x^{41}+x^{40}\right)$$

$$+x^{38}+x^{36}+x^{34}+x^{33}+x^{32}+x^{31}+x^{30}+x^{27}+x^{26}+x^{25}+x^{22}$$
$$+x^{21}+x^{20}+x^{19}+x^{14}+x^{12}+x^{11}+x^{10}+x^{8}+x^{3}+1\bigr) \bmod 2$$
$$\bigl(x^{59}+x^{58}+x^{55}+x^{54}+x^{53}+x^{52}+x^{50}+x^{49}+x^{48}+x^{47}+x^{45}+x^{44}$$
$$+x^{43}+x^{37}+x^{35}+x^{31}+x^{30}+x^{29}+x^{28}+x^{25}+x^{22}+x^{21}+x^{20}$$
$$+x^{15}+x^{14}+x^{7}+x^{6}+x^{4}+x^{3}+x+1\bigr) \bmod 2$$
$$\bigl(x^{62}+x^{60}+x^{58}+x^{57}+x^{55}+x^{54}+x^{52}+x^{51}+x^{50}+x^{49}+x^{48}+x^{47}$$
$$+x^{46}+x^{43}+x^{42}+x^{41}+x^{40}+x^{39}+x^{34}+x^{33}+x^{32}+x^{29}+x^{20}$$
$$+x^{19}+x^{16}+x^{15}+x^{14}+x^{12}+x^{7}+x^{6}+1\bigr) \bmod 2$$
$$\bigl(x^{60}+x^{59}+x^{58}+x^{52}+x^{51}+x^{50}+x^{49}+x^{46}+x^{45}+x^{44}+x^{43}+x^{40}$$
$$+x^{38}+x^{37}+x^{34}+x^{33}+x^{31}+x^{28}+x^{27}+x^{25}+x^{24}+x^{23}+x^{20}$$
$$+x^{18}+x^{16}+x^{15}+x^{13}+x^{10}+x^{9}+x^{8}+x^{7}+x^{6}+x^{3}+x^{2}$$
$$+x\bigr) \bmod 2$$
$$\bigl(x^{55}+x^{54}+x^{53}+x^{52}+x^{50}+x^{48}+x^{46}+x^{41}+x^{38}+x^{37}+x^{28}+x^{26} \qquad (9.3.22.12)$$
$$+x^{25}+x^{24}+x^{17}+x^{16}+x^{15}+x^{14}+x^{13}+x^{12}+x^{11}+x^{8}+x^{7}+x^{4}$$
$$+1\bigr) \bmod 2$$

```
> CRCcodeweigth:=proc(k::posint,crcpoly::modp1) local i,l,L,
  m,s,w;
  m:=modp1(Degree(crcpoly),2); w:=k+m;
  for i to 2^k-1 do
    L:=convert(i,base,2); [0$'i'=1..m,op(L)]; modp1(ConvertIn
  (%,x),2);
    modp1(Multiply(%,crcpoly),2); L:=modp1(ConvertOut(%),2);
    s:=convert(%,`+`); if s<w then w:=s fi;
  od; w; end;
```

CRCcodeweigth := **proc**(*k*::*posint*, *crcpoly*::*modp1*) (9.3.22.13)

local *i*, *l*, *L*, *m*, *s*, *w*;
m := *modp1*(*Degree*(*crcpoly*), 2);
w := *k* + *m*;
for *i* **to** 2^*k* − 1 **do**
L := *convert*(*i*, *base*, 2);
[$(0, 'i' = 1..m), op(L)];
modp1(*ConvertIn*(%, *x*), 2);
modp1(*Multiply*(%, *crcpoly*), 2);
L := *modp1*(*ConvertOut*(%), 2);
s := *convert*(%, +);
if *s* < *w* **then**
w := *s*
end if

```
    end do;
    w
end proc
```

```
> CRCcodeweigth(4,CRC32);
```

$$15 \tag{9.3.22.14}$$

```
> CRCcodeweigth(8,CRC32);
```

$$15 \tag{9.3.22.15}$$

```
> CRCcodeweigth(12,CRC32);
```

$$12 \tag{9.3.22.16}$$

```
> CRCcodeweigth(16,CRC32);
```

$$11 \tag{9.3.22.17}$$

```
> CRCcodeweigth(20,CRC32);
```

$$11 \tag{9.3.22.18}$$

```
> CRCcodeweigth(4,CRC64ISO);
```

$$5 \tag{9.3.22.19}$$

```
> CRCcodeweigth(8,CRC64ISO);
```

$$5 \tag{9.3.22.20}$$

```
> CRCcodeweigth(12,CRC64ISO);
```

$$5 \tag{9.3.22.21}$$

```
> CRCcodeweigth(16,CRC64ISO);
```

$$5 \tag{9.3.22.22}$$

```
> CRCcodeweigth(20,CRC64ISO);
```

$$5 \tag{9.3.22.23}$$

*9.3.23. A négy Golay-kód.

```
> Golay2:=modp1(ConvertIn(x^11+x^10+x^6+x^5+x^4+x^2+1,x),2);
```

$$Golay2 := \left(x^{11} + x^{10} + x^6 + x^5 + x^4 + x^2 + 1\right) \bmod 2 \tag{9.3.23.1}$$

```
> codedistribution:=proc(p::posint,k::posint,genpoly::modp1,
  y) local i,j,L,m,s,S;
  m:=modp1(Degree(genpoly),p); S:=0;
  for i to p^k-1 do
    L:=convert(i,base,p); modp1(ConvertIn(%,x),p);
    modp1(Multiply(%,genpoly),p); L:=modp1(ConvertOut(%),p);
    [0$j=1..k+m-nops(L),op(L)];
    s:=map(j->y[j],%); s:=convert(%,`*`); S:=S+s;
  od; S; end;
```

$$codedistribution := \mathbf{proc}(p::posint,\ k::posint,\ genpoly::modp1,\ y) \tag{9.3.23.2}$$

```
    local i, j, L, m, s, S;
    m := modp1(Degree(genpoly), p);
    S := 0;
    for i to p^k - 1 do
```

```
        L := convert(i, base, p);
        modp1(ConvertIn(%, x), p);
        modp1(Multiply(%, genpoly), p);
        L := modp1(ConvertOut(%), p);
        [$(0, j = 1..k + m - nops(L)), op(L)];
        s := map(proc(j)
            option operator, arrow;
            y[j]
        end proc, %);
        s := convert(%, *);
        S := S + s
    end do;
    S
end proc
```

```
> codedistribution(2,12,Golay2,`y`);
```

$$253\,y_0^{16}y_1^7 + 506\,y_0^{15}y_1^8 + 1288\,y_0^{11}y_1^{12} + 1288\,y_0^{12}y_1^{11} + 506\,y_0^8y_1^{15} + 253\,y_0^7y_1^{16} + y_1^{23} \tag{9.3.23.3}$$

```
> alias(alpha=RootOf(x^5+2*x+1));
```

$$\alpha \tag{9.3.23.4}$$

```
> beta:=Expand(alpha^22) mod 3;
```

$$\beta := 2\,\alpha^3 + 2\,\alpha^2 + \alpha + 1 \tag{9.3.23.5}$$

```
> p:=Expand((x-beta)*(x-beta^3)*(x-beta^4)*(x-beta^5)*(x-
  beta^9)) mod 3;
```

$$p := x^5 + 2\,x^3 + x^4 + 2 + x^2 \tag{9.3.23.6}$$

```
> Golay3:=modp1(ConvertIn(p,x),3);
```

$$Golay3 := \left(x^5 + x^4 + 2\,x^3 + x^2 + 2\right) \bmod 3 \tag{9.3.23.7}$$

```
> codedistribution(3,6,Golay3,`y`);
```

$$55\,y_0^6y_2^2y_1^3 + 55\,y_0^6y_1^2y_2^3 + 110\,y_0^5y_1^3y_2^3 + 11\,y_0^5y_1^6 + 110\,y_0^3y_1^3y_2^5 + 11\,y_0^6y_1^5 + 11\,y_0^5y_2^6 + 11\,y_0^6y_2^5 + 110\,y_0^3y_2^3y_1^5 + 55\,y_0^3y_1^6y_2^2 + 55\,y_0^2y_1^6y_2^3 + 55\,y_0^3y_2^6y_1^2 + 55\,y_0^2y_2^6y_1^3 + 11\,y_1^6y_2^5 + 11\,y_1^5y_2^6 + y_1^{11} + y_2^{11} \tag{9.3.23.8}$$

▶ 9.3.24. Vandermonde-determináns.

▼ 9.3.25. Reed-Solomon-kódok.

```
> with(PolynomialTools);
```

[*CoefficientList*, *CoefficientVector*, *GcdFreeBasis*, (9.3.25.1)

GreatestFactorialFactorization, Hurwitz, IsSelfReciprocal, MinimalPolynomial, PDEToPolynomial, PolynomialToPDE, ShiftEquivalent, ShiftlessDecomposition, Shorten, Shorter, Sort, Split, Splits, Translate]

```
> lgn:=8; n:=2^lgn-1; M:=Nextprime(Z^lgn,Z) mod 2; alpha:=
  Z+1;
```

$$lgn := 8$$
$$n := 255$$
$$M := Z^8 + Z^4 + Z^3 + Z + 1$$
$$\alpha := Z + 1 \qquad (9.3.25.2)$$

```
> ordermodM:=proc(M,Z,lgn,alpha) local i,beta; beta:=1;
  for i to 2^lgn-1 do
    beta:=modpol(beta*alpha,M,Z,2);
    if beta=1 then return i fi;
  od; FAIL end;

  ordermodM(M,Z,lgn,alpha);
```

```
ordermodM := proc(M, Z, lgn, α)
    local i, β;
    β := 1;
    for i to 2^lgn − 1 do
        β := modpol(β*α, M, Z, 2);
        if β = 1 then
            return i
        end if
    end do;
    FAIL
end proc
```

$$255 \qquad (9.3.25.3)$$

```
> m:=16; g:=mul((z-alpha^i),i=1..m);
```

$$m := 16$$
$$g := (z - Z - 1)\,(z - (Z+1)^2)\,(z - (Z+1)^3)\,(z - (Z+1)^4)\,(z - (Z+1)^5)\,(z - (Z+1)^6)\,(z - (Z+1)^7)\,(z - (Z+1)^8)\,(z - (Z+1)^9)\,(z - (Z+1)^{10})\,(z - (Z+1)^{11})\,(z - (Z+1)^{12})\,(z - (Z+1)^{13})\,(z - (Z+1)^{14})\,(z - (Z+1)^{15})\,(z - (Z+1)^{16}) \qquad (9.3.25.4)$$

```
> normalizepolyzZ:=proc(p,M)
  local i;
  CoefficientList(expand(p) mod 2,z);
```

```
    map(x->modpol(x,M,Z,2),%);
    add(%[i]*z^(i-1),i=1..nops(%)); sort(%); end;
```

$$normalizepolyzZ := \mathbf{proc}(p, M) \tag{9.3.25.5}$$

```
    local i;
    PolynomialTools:-CoefficientList(mod(expand(p), 2), z);
    map(proc(x)
        option operator, arrow;
        modpol(x, M, Z, 2)
    end proc, %);
    add(%[i]*z^(i-1), i=1..nops(%));
    sort(%)
end proc
```

```
> g:=normalizepolyzZ(g,M);
```

$$g := z^{16} + \left(Z^6 + Z^5 + Z^4 + 1\right) z^{15} + \left(Z^5 + Z^2 + Z + 1\right) z^{14} + \left(Z^6 + Z^5 + 1\right) z^{13} + \left(Z^4 + Z^3 + 1\right) z^{12} + \left(Z^6 + Z^5 + Z^3 + 1\right) z^{11} + z^7 Z^4 + \left(Z^6 + Z^5 + Z^4 + Z^3 + Z^2 + Z\right) z^{10} + \left(Z^7 + Z^6 + Z^4\right) z^9 + \left(Z^7 + Z^4 + Z^2 + Z\right) z^8 + \left(Z^5 + Z^3 + Z^2 + 1\right) z^6 + Z^6 + \left(Z^7 + Z^6 + Z^4 + Z^3 + Z\right) z^5 + \left(Z^5 + Z + 1\right) z^4 + Z^4 + \left(Z^6 + Z^5 + Z + 1\right) z^3 + Z^3 + \left(Z^6 + Z^2 + 1\right) z^2 + Z^2 + \left(Z^5 + Z^4 + Z^3 + 1\right) z + 1 \tag{9.3.25.6}$$

```
> rnd:=rand(2^lgn): k:=n-m; mv:=Vector(k); for i to k do mv
  [i]:=rnd(i) od:
  mv[1]; mv:=map(x->convert(x,base,2),mv): mv[1]; i:='i':
  mv:=map(x->sum(x[i]*Z^(i-1),i=1..nops(x)),mv): mv[1];
```

$$k := 239$$

$$mv := \begin{bmatrix} 1 \,..\, 239\ Vector_{column} \\ Data\ Type: anything \\ Storage: rectangular \\ Order: Fortran\_order \end{bmatrix}$$

$$225$$

$$[1, 0, 0, 0, 0, 1, 1, 1]$$

$$1 + Z^5 + Z^6 + Z^7 \tag{9.3.25.7}$$

```
> c:=add(mv[i]*z^(i-1),i=1..k)*g;
```

$$c := \left(1 + \left(Z^2 + Z^5 + Z^7\right) z^{186} + \left(1 + Z + Z^2 + Z^4 + Z^6\right) z^{187} + \left(Z^4 + Z^5 + Z^6 + Z^7\right) z^{188} + \left(1 + Z^4 + Z^5 + Z^7\right) z^{189} + \left(Z^3 + Z^5 + Z^6\right) z^{190} + \left(Z^2 + Z^5\right) z^{191} + \left(Z^7 + Z^6 + Z^3 + Z + 1\right) z^{192} + \left(Z^6 + Z^4 + Z^3 + Z^2\right) \tag{9.3.25.8}$$

$$+ Z\big) z^{193} + \big(1 + Z + Z^2 + Z^7\big) z^{194} + \big(1 + Z + Z^3 + Z^4 + Z^6\big) z^{195} + \big(Z^2 + Z^3 + Z^4 + Z^5 + Z^6 + Z^7\big) z^{196} + \big(1 + Z^7 + Z^6 + Z^4 + Z^3 + Z\big) z^{197} + \big(1 + Z^7\big) z^{198} + \big(Z^7 + Z^5 + Z^4 + Z + 1\big) z^{94} + \big(Z^5 + Z^4 + Z^3 + 1\big) z^{96} + \big(Z + Z^6\big) z^{97} + \big(1 + Z^4 + Z^5 + Z^6 + Z^7\big) z^{98} + \big(Z^3 + Z^4 + Z^5 + Z^6\big) z^{99} + \big(Z^7 + Z^6 + Z^4\big) z^{100} + Z^6 z^{101} + \big(Z^2 + Z^6 + Z^7\big) z^{102} + \big(1 + Z^7\big) z^{103} + \big(Z^5 + Z + 1\big) z^{104} + \big(Z^2 + Z^3 + Z^4 + Z^5 + Z^7\big) z^{105} + \big(1 + Z + Z^4 + Z^6\big) z^{106} + \big(1 + Z + Z^2 + Z^3 + Z^4 + Z^6 + Z^7\big) z^{107} + \big(1 + Z^6 + Z^5 + Z^4 + Z^3 + Z^2 + Z\big) z^{95} + \big(Z^2 + Z^5 + Z^7\big) z^{71} + \big(1 + Z + Z^2 + Z^3 + Z^5 + Z^6\big) z^{72} + \big(Z + Z^3 + Z^6\big) z^{73} + \big(Z + 1\big) z^{74} + \big(Z^7 + Z^6 + Z^4 + Z^3 + Z\big) z^{75} + \big(1 + Z^7 + Z^4 + Z^2\big) z^{76} + \big(Z^5 + Z^2 + Z + 1\big) z^{77} + \big(Z^5 + Z^4 + Z^3 + 1\big) z^{78} + \big(Z^2 + Z^3 + Z^4 + Z^6 + Z^7\big) z^{79} + \big(1 + Z^5 + Z^6 + Z^7\big) z^{80} + \big(Z + Z^3 + Z^5 + Z^6\big) z^{81} + \big(Z + Z^3\big) z^{82} + \big(Z^4 + Z^3 + Z^2 + 1\big) z^{83} + \big(1 + Z^2 + Z^7\big) z^{84} + \big(1 + Z + Z^2 + Z^3 + Z^4\big) z^{85} + \big(Z^5 + Z^2 + Z + 1\big) z^{86} + \big(1 + Z^6 + Z^5 + Z^4 + Z\big) z^{87} + \big(1 + Z^7 + Z^4 + Z^2 + Z\big) z^{88} + \big(1 + Z + Z^2 + Z^3 + Z^6\big) z^{89} + \big(1 + Z^6 + Z^4 + Z^3 + Z^2 + Z\big) z^{90} + \big(Z + Z^3\big) z^{91} + \big(Z + Z^2 + Z^4\big) z^{92} + \big(Z^3 + Z^5 + Z^7\big) z^{93} + \big(Z^5 + Z^6\big) z^{199} + \big(1 + Z + Z^3 + Z^5\big) z^{200} + Z^3 z^{201} + \big(Z^4 + Z^3 + Z^2 + 1\big) z^{202} + \big(Z^7 + Z^3 + Z^2 + 1\big) z^{203} + \big(Z^6 + Z^5 + Z^3 + Z^2\big) z^{204} + \big(Z + Z^2 + Z^3 + Z^6\big) z^{205} + \big(Z + Z^2 + Z^4 + Z^5 + Z^6 + Z^7\big) z^{206} + \big(Z^4 + Z^5 + Z^6 + Z^7\big) z^{207} + \big(Z^3 + Z^6\big) z^{208} + \big(Z^2 + Z^4\big) z^{209} + \big(Z^6 + Z^3 + Z^2 + 1\big) z^{210} + \big(Z^4 + Z^5\big) z^{211} + \big(1 + Z^7 + Z^6 + Z^5 + Z^4 + Z^3 + Z\big) z^{212} + \big(Z^2 + Z^3 + Z^4 + Z^6 + Z^7\big) z^{213} + \big(1 + Z^6 + Z^5 + Z^4 + Z\big) z^{214} + \big(Z + Z^2 + Z^3\big) z^{215} + \big(1 + Z^5 + Z^4 + Z^2 + Z\big) z^{216} + \big(Z^2 + Z^6\big) z^{217} + \big(Z^4 + Z^7\big) z^{218} + \big(Z^7 + Z^4 + Z^2\big) z^{219} + \big(1 + Z^3 + Z^5\big) z^{220} + \big(Z^7 + Z^4 + Z^2\big) z^{221} + \big(Z^2 + Z^5 + Z^6\big) z^{222} + \big(1 + Z^4 + Z^3 + Z^2 + Z^6\big) z^{150} + \big(Z^7 + Z^5 + Z^4 + Z + 1\big) z^{151} + \big(1 + Z + Z^2 + Z^3\big) z^{152} + \big(Z^2 + Z^3 + Z^4 + Z^7\big) z^{153} + \big(Z + Z^4 + Z^5\big) z^{154} + \big(Z + Z^7\big) z^{155} + \big(Z^7 + Z^6 + Z^5 + Z^3\big) z^{156} + \big(Z^2 + Z^3 + Z^5 + Z^7\big) z^{157} + \big(Z^7 + Z^6 + Z\big) z^{158} + \big(1 + Z + Z^2 + Z^5 + Z^7\big) z^{159} + \big(Z + Z^2 + Z^6\big) z^{160} + \big(Z^3 + Z^5 + Z^6\big) z^{161} + \big(1 + Z^3 + Z^6\big) z^{162} + \big(Z^4 + Z^5\big) z^{163} + \big(1 + Z + Z^3 + Z^4 + Z^5 + Z^7\big) z^{164} + \big(Z^2 + Z^6 + Z^7\big) z^{165} + \big(Z^3 + Z^4 + Z^5 + Z^6 + Z^7\big) z^{166} + Z^5 z^{167} + \big(1 + Z^2 + Z^3 + Z^4 + Z^7\big) z^{168} + Z^7 z^{169} + \big(Z + Z^4 + Z^6 + Z^7\big) z^{170} + \big(Z^3 + Z^7\big) z^{171} + \big(Z^3 + Z^5 + Z^6\big) z^{172} + \big(Z^7 + Z^6 + Z^5$$

$$+ Z^4 + Z^3 + Z\big) z^{173} + \big(1 + Z + Z^2 + Z^3 + Z^4 + Z^5\big) z^{174} + \big(Z + Z^3 + Z^5\big) z^{175} + \big(Z^3 + Z^4 + Z^5 + Z^7\big) z^{176} + \big(Z^7 + Z^5 + Z^4 + Z + 1\big) z^{177} + \big(Z^7 + Z^5 + Z^4 + Z^3 + Z^2 + 1\big) z^{178} + \big(1 + Z^2 + Z^3 + Z^5 + Z^7\big) z^{179} + \big(Z^6 + Z^5 + Z^4 + Z^3 + Z^2 + Z\big) z^{180} + \big(Z^6 + Z^5 + Z^3 + Z^2\big) z^{181} + \big(Z^7 + Z^6 + Z^3 + Z + 1\big) z^{182} + Z^3 z^{183} + \big(1 + Z^4 + Z^5 + Z^6 + Z^7\big) z^{184} + \big(Z^2 + Z^4 + Z^5 + Z^7\big) z^{185} + Z^2 z^{223} + \big(1 + Z^3 + Z^4 + Z^5 + Z^7\big) z^{224} + Z z^{225} + \big(1 + Z^2 + Z^4 + Z^6\big) z^{226} + \big(1 + Z^2 + Z^4 + Z^5 + Z^6 + Z^7\big) z^{227} + \big(Z + Z^2 + Z^4 + Z^5 + Z^7\big) z^{228} + \big(Z^5 + Z^6 + Z^7\big) z^{229} + \big(Z + 1\big) z^{230} + \big(1 + Z + Z^3 + Z^4 + Z^6\big) z^{231} + \big(Z^5 + Z^3 + Z^2 + 1\big) z^{232} + \big(1 + Z^4 + Z^5 + Z^6 + Z^7\big) z^{233} + \big(1 + Z + Z^2 + Z^4 + Z^6 + Z^7\big) z^{234} + \big(1 + Z^5 + Z^7\big) z^{235} + \big(1 + Z^4 + Z^5 + Z^6 + Z^7\big) z^{236} + \big(1 + Z^7 + Z^6 + Z\big) z^{237} + \big(Z^4 + Z^5\big) z^{238} + Z^5 + \big(1 + Z^2 + Z^3 + Z^4 + Z^6 + Z^7\big) z^{108} + Z^6 + \big(Z + Z^2 + Z^3 + Z^4 + Z^7\big) z^{109} + \big(Z^7 + Z^6 + Z^4\big) z^{110} + \big(Z + Z^2 + Z^3 + Z^4\big) z^{111} + \big(1 + Z^2 + Z^4 + Z^5\big) z^{112} + \big(Z + Z^6\big) z^{113} + \big(Z^4 + Z^5 + Z^6 + Z^7\big) z^{114} + \big(Z + Z^3 + Z^4 + Z^5 + Z^7\big) z^{115} + \big(Z + Z^2 + Z^5 + Z^6 + Z^7\big) z^{116} + \big(1 + Z^6\big) z^{117} + \big(1 + Z + Z^3 + Z^7\big) z^{118} + \big(Z^4 + Z^5 + Z^7\big) z^{119} + \big(Z^2 + Z^5 + Z^6\big) z^{120} + \big(1 + Z^3 + Z^4 + Z^6\big) z^{121} + \big(Z + Z^5\big) z^{122} + \big(Z + Z^2 + Z^3\big) z^{123} + \big(1 + Z^3 + Z^4 + Z^7\big) z^{124} + \big(Z^6 + Z^5 + Z + 1\big) z^{125} + \big(1 + Z^2 + Z^4 + Z^5 + Z^6\big) z^{126} + \big(1 + Z^7 + Z^4 + Z^2 + Z\big) z^{127} + \big(1 + Z + Z^2 + Z^7\big) z^{128} + \big(Z + Z^2 + Z^3 + Z^4 + Z^5 + Z^7\big) z^{129} + \big(Z^6 + Z^3 + Z^2 + 1\big) z^{130} + \big(Z^7 + Z^4 + Z^2 + Z\big) z^{131} + \big(Z^3 + Z^6\big) z^{133} + \big(1 + Z^7 + Z^4 + Z^2\big) z^{134} + \big(1 + Z^2 + Z^3 + Z^5 + Z^6 + Z^7\big) z^{132} + \big(1 + Z + Z^2 + Z^6\big) z + \big(1 + Z^6 + Z^5 + Z^4 + Z\big) z^2 + \big(Z + Z^2 + Z^3 + Z^5 + Z^6 + Z^7\big) z^3 + \big(1 + Z^7\big) z^4 + \big(Z^3 + Z^6\big) z^5 + \big(Z^2 + Z^4 + Z^6\big) z^6 + \big(Z^3 + Z^5\big) z^7 + \big(Z + Z^2 + Z^5 + Z^6 + Z^7\big) z^8 + \big(Z + 1\big) z^9 + \big(1 + Z + Z^3 + Z^5 + Z^7\big) z^{10} + \big(Z^4 + Z^5 + Z^6 + Z^7\big) z^{135} + \big(1 + Z^5 + Z^6 + Z^7\big) z^{137} + Z^3 z^{138} + \big(Z^2 + Z^4 + Z^5\big) z^{139} + Z z^{140} + \big(Z^6 + Z^5 + 1\big) z^{141} + \big(Z^2 + Z^6\big) z^{142} + \big(Z^2 + Z^3 + Z^4 + Z^5 + Z^6\big) z^{143} + \big(Z + Z^7\big) z^{144} + \big(1 + Z + Z^2 + Z^4 + Z^6 + Z^7\big) z^{146} + \big(Z^2 + Z^5 + Z^7\big) z^{147} + \big(1 + Z + Z^3 + Z^4 + Z^5 + Z^7\big) z^{148} + \big(1 + Z^6 + Z^5 + Z^3 + Z^2\big) z^{149} + \big(1 + Z + Z^2 + Z^3 + Z^4 + Z^7\big) z^{136} + \big(Z^2 + Z^3\big) z^{11} + \big(Z + Z^4 + Z^7\big) z^{12} + \big(1 + Z^3\big) z^{13} + \big(Z + Z^2 + Z^3 + Z^7\big) z^{14} + \big(1 + Z + Z^2 + Z^3 + Z^4 + Z^5 + Z^7\big) z^{15} + Z^3 z^{16} + \big(1 + Z^3 + Z^4 + Z^5 + Z^6 + Z^7\big) z^{17} + \big(1 + Z + Z^3\big) z^{18} + \big(Z^2 + Z^3 + Z^4 + Z^5 + Z^6 + Z^7\big) z^{19}$$

$$+\left(Z+Z^3+Z^5+Z^6+Z^7\right)z^{20}+\left(1+Z+Z^4+Z^5\right)z^{21}+\left(1+Z+Z^2+Z^3+Z^4+Z^5+Z^6+Z^7\right)z^{22}+\left(Z^3+Z^4+Z^7\right)z^{23}+\left(Z+Z^2+Z^4+Z^5+Z^6+Z^7\right)z^{24}+\left(Z^3+Z^5+Z^6\right)z^{26}+\left(Z^4+Z^5+Z^6\right)z^{27}+\left(1+Z^3+Z^4+Z^6\right)z^{28}+\left(1+Z^4\right)z^{29}+\left(Z+Z^3+Z^5+Z^7\right)z^{30}+\left(1+Z^6+Z^5+Z^4+Z\right)z^{31}+\left(Z+Z^2+Z^4+Z^5+Z^7\right)z^{32}+\left(Z^6+Z^5+Z^3+Z^2\right)z^{33}+\left(Z^5+Z^2+Z+1\right)z^{34}+Z^4z^{35}+\left(1+Z^2+Z^6+Z^7\right)z^{36}+\left(1+Z+Z^4+Z^5\right)z^{37}+\left(Z^4+Z^6\right)z^{38}+\left(Z+Z^2+Z^4+Z^6+Z^7\right)z^{39}+\left(1+Z^5+Z^7\right)z^{40}+\left(Z^3+Z^4+Z^5+Z^6+Z^7\right)z^{41}+\left(Z^2+Z^4+Z^6+Z^7\right)z^{42}+\left(Z^3+Z^7\right)z^{43}+\left(Z^3+Z^5+Z^6\right)z^{44}+\left(1+Z^7+Z^6+Z\right)z^{45}+\left(Z+Z^4+Z^6\right)z^{46}+\left(Z^2+Z^3+Z^4+Z^5+Z^6\right)z^{47}+\left(1+Z+Z^2+Z^6\right)z^{25}+\left(Z^3+Z^6\right)z^{145}+Z^7+\left(Z+Z^2+Z^3+Z^4+Z^5+Z^7\right)z^{48}+\left(1+Z^2+Z^6+Z^7\right)z^{50}+\left(Z^5+Z+1\right)z^{51}+\left(1+Z+Z^5+Z^7\right)z^{52}+\left(1+Z^3\right)z^{53}+\left(Z^5+Z^2+Z+1\right)z^{54}+\left(1+Z^6\right)z^{55}+\left(Z+Z^2+Z^4+Z^6\right)z^{56}+\left(1+Z+Z^2+Z^3+Z^7\right)z^{57}+\left(Z^6+Z^5+1\right)z^{58}+\left(Z+Z^3+Z^5+Z^7\right)z^{59}+\left(Z^2+Z^4+Z^5\right)z^{60}+\left(Z^2+Z^3+Z^4+Z^5\right)z^{61}+\left(Z+Z^2+Z^3+Z^5\right)z^{62}+\left(1+Z+Z^3+Z^4\right)z^{63}+\left(Z+Z^3+Z^6+Z^7\right)z^{64}+Zz^{65}+\left(Z^2+Z^7\right)z^{66}+\left(1+Z^6+Z^5+Z^3+Z^2\right)z^{67}+\left(Z+Z^2+Z^5+Z^6+Z^7\right)z^{68}+\left(Z+Z^3+Z^5+Z^6\right)z^{69}+\left(Z+Z^3\right)z^{70}+\left(Z^6+Z^5+Z+1\right)z^{49}\left(z^{16}+\left(Z^6+Z^5+Z^4+1\right)z^{15}+\left(Z^5+Z^2+Z+1\right)z^{14}+\left(Z^6+Z^5+1\right)z^{13}+\left(Z^4+Z^3+1\right)z^{12}+\left(Z^6+Z^5+Z^3+1\right)z^{11}+z^7Z^4+\left(Z^6+Z^5+Z^4+Z^3+Z^2+Z\right)z^{10}+\left(Z^7+Z^6+Z^4\right)z^9+\left(Z^7+Z^4+Z^2+Z\right)z^8+\left(Z^5+Z^3+Z^2+1\right)z^6+Z^6+\left(Z^7+Z^6+Z^4+Z^3+Z\right)z^5+\left(Z^5+Z+1\right)z^4+Z^4+\left(Z^6+Z^5+Z+1\right)z^3+Z^3+\left(Z^6+Z^2+1\right)z^2+Z^2+\left(Z^5+Z^4+Z^3+1\right)z+1\right)$$

```
> c:=normalizepolyzZ(c,M);
```

$$c:=\left(Z^5+Z^4\right)z^{254}+\left(Z^5+Z^4\right)z^{253}+\left(Z^6+Z^3+1\right)z^{252}+\left(Z^6+Z^5+Z^4\right)z^{251}+\left(Z^5+Z\right)z^{250}+\left(Z^3+Z\right)z^{249}+\left(Z^7+Z^4+Z^2+Z\right)z^{248}+\left(Z^7+Z^4+Z^3+Z^2+Z+1\right)z^{247}+\left(Z^7+Z^4+Z^3\right)z^{246}+\left(Z^7+Z^6+Z^5+Z^4+Z^3+Z^2+1\right)z^{245}+\left(Z^6+Z^5+Z^2+Z+1\right)z^{244}+\left(Z^3+1\right)z^{243}+\left(Z^3+Z^2+Z\right)z^{242}+z^{236}Z^6+\left(Z^6+Z^4\right)z^{241}+\left(Z^5+1\right)z^{240}+\left(Z^6+Z^5+Z^3+1\right)z^{239}+\left(Z^7+Z^5+Z^3+Z^2+Z\right)z^{238}+\left(Z^6+Z^5+Z^3+1\right)z^{237}+\left(Z^3+Z\right)z^{235}+\left(Z^6+Z^5+Z^2+Z\right)z^{234}+\left(Z+1\right)z^{233}+z^{232}Z+\left(Z^5+Z^3+Z+1\right)z^{231} \tag{9.3.25.9}$$

$$+\left(Z^5+Z+1\right)z^{230}+z^{229}Z+\left(Z^7+Z^6+Z^5\right)z^{228}+\left(Z^7+Z^6+Z^4\right.$$
$$\left.+Z^3+Z^2\right)z^{227}+\left(Z^3+Z^2+1\right)z^{226}+\left(Z^6+Z^5+Z^4+Z^2+Z\right.$$
$$\left.+1\right)z^{225}+\left(Z^6+Z^4+Z^3+Z\right)z^{224}+\left(Z^2+1\right)z^{223}+\left(Z^6+1\right)z^{222}$$
$$+\left(Z^7+Z^3+1\right)z^{221}+\left(Z^5+Z^4+Z^3\right)z^{220}+\left(Z^5+Z^4+Z^3\right.$$
$$\left.+Z^2\right)z^{219}+\left(Z^6+Z^4+Z^2+1\right)z^{218}+\left(Z^7+Z^6+Z^4+Z^3+1\right)z^{217}$$
$$+\left(Z^6+Z^5+Z^3+Z+1\right)z^{216}+\left(Z^6+Z^5+Z^4\right)z^{215}+\left(Z^7+Z^6\right.$$
$$\left.+Z^4\right)z^{214}+\left(Z^7+Z^5+Z^3+Z^2\right)z^{213}+\left(Z^7+Z^6+Z^4+Z^3+1\right)z^{212}$$
$$+\left(Z^7+1\right)z^{211}+\left(Z^5+Z^4+Z\right)z^{210}+\left(Z^7+Z^5+Z^3+Z^2+Z\right.$$
$$\left.+1\right)z^{209}+\left(Z^5+Z^4+Z\right)z^{208}+\left(Z^5+Z^3+1\right)z^{207}+\left(Z^7+Z^4\right.$$
$$\left.+1\right)z^{206}+\left(Z^7+Z^6+Z^3+Z^2+1\right)z^{205}+\left(Z^7+Z^6+Z^5+Z^4+Z^3\right.$$
$$\left.+Z+1\right)z^{204}+\left(Z^7+Z^5+Z^3+Z+1\right)z^{203}+\left(Z^7+Z^2+Z\right)z^{201}$$
$$+\left(Z^3+1\right)z^{200}+\left(Z^7+Z^3+Z\right)z^{199}+\left(Z^6+Z^3+Z^2+1\right)z^{198}$$
$$+\left(Z^7+Z^6+Z^3+Z^2+Z\right)z^{197}+\left(Z^5+Z^4+Z^3+1\right)z^{196}+\left(Z^7+Z^6\right.$$
$$\left.+Z^4+Z^2+1\right)z^{195}+\left(Z^7+Z^3+Z\right)z^{194}+\left(Z^4+1\right)z^{193}+\left(Z^6+Z^4\right.$$
$$\left.+Z^2+1\right)z^{192}+\left(Z^7+Z^4+Z^2\right)z^{191}+\left(Z^6+Z^5+Z^4+Z^3+Z^2\right)z^{190}$$
$$+\left(Z^7+Z^5+Z^4+Z^3+Z+1\right)z^{189}+\left(Z^5+Z^4+Z^3+1\right)z^{188}+\left(Z^6\right.$$
$$\left.+Z^5+Z^4+Z^3+1\right)z^{187}+\left(Z^5+Z^4+Z^3\right)z^{186}+\left(Z^7+Z^6+Z^5\right.$$
$$\left.+1\right)z^{185}+\left(Z^5+Z+1\right)z^{184}+\left(Z^7+Z^6+Z^4+Z^3+Z^2+1\right)z^{183}$$
$$+\left(Z^7+Z^6+Z^5+Z^3+1\right)z^{182}+\left(Z+1\right)z^{181}+\left(Z^4+Z^2+Z\right.$$
$$\left.+1\right)z^{180}+\left(Z^7+Z^2+Z\right)z^{179}+\left(Z^7+Z^6+Z^3+Z^2+Z+1\right)z^{178}$$
$$+\left(Z^6+Z^4+Z^3+Z^2+Z\right)z^{177}+\left(Z^7+Z^4+Z^3+Z^2\right)z^{176}+\left(Z^6\right.$$
$$\left.+Z^5+Z\right)z^{175}+\left(Z^7+Z^3+1\right)z^{174}+\left(Z^6+Z^5+Z^2+Z+1\right)z^{173}$$
$$+\left(Z^7+Z^6+Z^5+Z^2\right)z^{172}+\left(Z^6+Z^4+Z^3\right)z^{171}+\left(Z^5+Z^4\right)z^{170}$$
$$+\left(Z^7+Z^6+Z^5+Z^3\right)z^{169}+\left(Z^7+1\right)z^{168}+\left(Z^7+Z^5+Z^3\right.$$
$$\left.+Z\right)z^{167}+\left(Z^3+Z^2\right)z^{166}+\left(Z^7+Z^3+1\right)z^{165}+\left(Z^6+Z^5+Z^2+Z\right.$$
$$\left.+1\right)z^{164}+\left(Z^7+Z^3\right)z^{163}+\left(Z^7+Z^5+Z^4+Z^2\right)z^{162}+\left(Z^7+Z^4\right.$$
$$\left.+Z^3+Z^2+Z\right)z^{161}+\left(Z^5+Z^4+Z^3+1\right)z^{160}+\left(Z^6+Z^4+Z^3+Z^2\right.$$
$$\left.+Z\right)z^{159}+\left(Z^7+Z^4\right)z^{158}+\left(Z^7+Z^5+Z^3+Z^2\right)z^{157}+\left(Z^7+Z^6\right.$$
$$\left.+Z^4+Z^3+Z+1\right)z^{156}+\left(Z^4+Z^3+Z+1\right)z^{155}+\left(Z^7+Z^5\right.$$
$$\left.+Z^3\right)z^{154}+\left(Z^6+Z^5+Z^3+Z^2\right)z^{153}+\left(Z^7+Z^6+Z^4+Z\right)z^{152}$$
$$+\left(Z^7+Z^4+Z^3\right)z^{151}+\left(Z^6+Z^5+Z^4+Z+1\right)z^{150}+\left(Z^3\right.$$
$$\left.+Z^2\right)z^{149}+\left(Z^7+Z^6+Z^5+Z^3+Z+1\right)z^{148}+\left(Z^6+Z^5+1\right)z^{147}$$
$$+\left(Z^7+Z^6+Z^5\right)z^{146}+\left(Z^6+Z^5+Z^4+Z^2\right)z^{145}+\left(Z^6+Z^5+Z^3\right.$$
$$\left.+1\right)z^{144}+\left(Z^7+Z^6+Z^5+Z^4+Z^2+Z\right)z^{143}+\left(Z^7+Z^4+Z^2\right.$$

$$+ Z\big) z^{142} + \big(Z^4 + Z^2 + Z\big) z^{141} + \big(Z^3 + Z^2 + Z + 1\big) z^{140} + \big(Z^5 + Z^4 + Z^2 + 1\big) z^{139} + \big(Z^6 + Z^5 + Z^4 + 1\big) z^{138} + \big(Z^2 + Z\big) z^{137} + \big(Z^7 + Z^6 + Z^5 + 1\big) z^{136} + \big(Z^4 + Z^3 + Z^2 + 1\big) z^{135} + \big(Z^7 + Z^5 + Z^3 + Z^2\big) z^{134} + \big(Z^7 + Z^2 + Z\big) z^{133} + \big(Z^7 + Z^5 + 1\big) z^{132} + \big(Z^7 + Z^6 + Z^5 + Z^4 + Z^3 + Z^2 + 1\big) z^{131} + \big(Z^7 + Z^5 + Z^4 + Z^3 + Z + 1\big) z^{130} + \big(Z^6 + Z^5 + Z^3 + 1\big) z^{129} + \big(Z^5 + Z^2 + Z + 1\big) z^{128} + \big(Z^3 + 1\big) z^{127} + \big(Z^6 + Z^5 + Z^4 + Z^3 + Z\big) z^{126} + \big(Z^6 + Z^5 + Z^3 + Z^2 + Z + 1\big) z^{125} + \big(Z^5 + Z^4 + Z^3 + Z^2\big) z^{124} + \big(Z^7 + Z^5 + Z^3 + Z^2 + Z + 1\big) z^{123} + \big(Z^7 + Z^5 + Z^4 + Z^3 + Z + 1\big) z^{122} + \big(Z^7 + Z^6 + Z^3 + Z^2 + 1\big) z^{121} + \big(Z^4 + Z^3 + Z^2 + 1\big) z^{120} + \big(Z^7 + Z^5 + 1\big) z^{119} + \big(Z^7 + Z^5 + Z^3 + Z^2 + 1\big) z^{118} + \big(Z^7 + Z^6 + Z^4 + Z^2\big) z^{117} + \big(Z^7 + Z^6 + Z^5 + 1\big) z^{116} + \big(Z^4 + Z + 1\big) z^{115} + \big(Z^7 + Z^4 + Z^3 + 1\big) z^{114} + \big(Z^4 + Z^2\big) z^{113} + \big(Z^4 + Z^2 + Z + 1\big) z^{112} + \big(Z^7 + Z^5 + Z^4 + Z^3 + Z + 1\big) z^{111} + \big(Z^6 + Z^4\big) z^{110} + \big(Z^4 + 1\big) z^{109} + \big(Z^6 + Z^5 + Z^4 + Z^3 + Z^2 + Z\big) z^{108} + \big(Z^4 + Z^3\big) z^{107} + \big(Z^6 + Z^5 + Z^4 + Z^3\big) z^{106} + \big(Z^7 + Z^5\big) z^{105} + \big(Z^7 + Z^6 + Z^4 + Z^3 + Z + 1\big) z^{104} + \big(Z^5 + Z^3 + 1\big) z^{103} + \big(Z^5 + Z^4 + Z + 1\big) z^{102} + z^{98} Z^4 + \big(Z^7 + Z^6 + Z^4 + Z^3 + 1\big) z^{101} + \big(Z^7 + Z^5 + Z^4 + Z^3 + Z^2 + Z\big) z^{100} + \big(Z^7 + Z^5 + Z^4 + Z^3 + Z^2\big) z^{99} + z^{93} Z^6 + \big(Z^6 + Z^5 + Z^4 + Z^2 + 1\big) z^{97} + \big(Z^4 + Z^3 + Z^2\big) z^{96} + \big(Z^6 + Z^5 + Z^3\big) z^{95} + \big(Z^7 + Z^6 + Z^5 + Z^4 + Z^2 + Z + 1\big) z^{94} + \big(Z^6 + Z^5 + Z^2\big) z^{92} + \big(Z^6 + Z^4 + Z^3 + Z^2 + Z + 1\big) z^{91} + \big(Z^6 + Z^5 + Z^4 + Z^3 + 1\big) z^{90} + \big(Z^5 + Z^3 + Z + 1\big) z^{89} + \big(Z^7 + Z^6 + 1\big) z^{88} + \big(Z^6 + Z^5 + Z^2\big) z^{87} + \big(Z^7 + Z^5 + Z^4 + Z^3 + Z^2\big) z^{86} + \big(Z^7 + Z^5 + Z^4 + Z^3 + Z^2 + 1\big) z^{85} + \big(Z^7 + Z^6 + Z^5 + Z^2 + Z + 1\big) z^{84} + \big(Z^5 + Z^3 + Z^2 + 1\big) z^{83} + \big(Z^7 + Z^6 + Z^5 + Z^2 + Z + 1\big) z^{82} + \big(Z^6 + Z^4 + Z + 1\big) z^{81} + z^{79} Z^2 + \big(Z^7 + Z^6 + Z^5 + Z^4 + Z^3 + Z + 1\big) z^{80} + \big(Z^5 + 1\big) z^{78} + \big(Z^7 + Z^6 + Z^4 + Z^3 + Z^2 + 1\big) z^{77} + \big(Z^7 + Z^6 + Z^5 + Z^4 + Z^2 + 1\big) z^{76} + \big(Z^7 + Z^6 + Z^2 + Z + 1\big) z^{75} + z^{70} Z^5 + \big(Z^7 + Z^5 + Z^4 + Z^3 + Z\big) z^{73} + \big(Z^7 + Z^6 + Z^5 + Z^4 + Z^2 + 1\big) z^{72} + \big(Z^6 + Z^2\big) z^{71} + \big(Z^6 + Z^5 + Z^3 + Z^2 + Z + 1\big) z^{69} + \big(Z^4 + Z^3 + Z^2 + Z\big) z^{68} + \big(Z^7 + Z^6 + Z^5 + Z^4 + Z^3 + Z^2 + Z\big) z^{67} + \big(Z^5 + Z^4 + Z^2\big) z^{66} + \big(Z^7 + Z^3 + Z^2 + Z\big) z^{65} + \big(Z^7 + Z^4 + Z^3 + Z^2\big) z^{64} + \big(Z^7 + Z^5 + 1\big) z^{63} + \big(Z^6 + Z^5 + Z^4 + Z^2 + Z\big) z^{62} + \big(Z^6 + Z^5 + Z^4 + Z^3 + 1\big) z^{61} + \big(Z^6 + Z^5 + Z^4 + Z^2 + 1\big) z^{60} + z^{59} + \big(Z^7 + Z^6 + Z^4 + Z^3 + Z\big) z^{58} + \big(Z^5 + Z^2 + Z + 1\big) z^{57} + \big(Z^2 + 1\big) z^{56} + \big(Z^5 + Z^4 + Z^3 + Z^2 + Z + 1\big) z^{55}$$

$$+ z^{52} Z^2 + \left(Z^5 + Z^4 + 1\right) z^{53} + \left(Z^5 + Z^4 + Z^3 + Z^2\right) z^{51} + \left(Z^7 + Z^6 + Z^5 + Z^3 + Z^2 + 1\right) z^{50} + \left(Z^7 + Z^5 + Z^4 + Z^3 + Z^2 + Z + 1\right) z^{49} + \left(Z^5 + Z^4 + Z^3 + Z^2\right) z^{48} + \left(Z^7 + Z^3 + Z^2 + Z\right) z^{47} + \left(Z^7 + Z^4\right) z^{46} + \left(Z^6 + Z^5 + Z^4 + Z^3 + Z + 1\right) z^{45} + \left(Z^5 + Z^4 + Z^3 + Z^2 + Z\right) z^{44} + \left(Z^2 + 1\right) z^{43} + \left(Z^7 + Z^4 + Z^2\right) z^{42} + \left(Z^7 + Z^4 + Z^2\right) z^{41} + \left(Z^6 + Z^5 + Z + 1\right) z^{40} + \left(Z^7 + Z^6 + Z^3 + Z^2\right) z^{39} + \left(Z^6 + Z^3\right) z^{38} + \left(Z^6 + Z^5 + Z^4 + Z^3 + Z^2 + 1\right) z^{37} + \left(Z^6 + Z^4 + Z^3 + Z + 1\right) z^{36} + \left(Z^6 + Z^5 + 1\right) z^{35} + \left(Z^7 + Z^6 + Z^4 + Z^2 + Z\right) z^{34} + \left(Z^6 + Z^5 + Z^4 + Z\right) z^{33} + \left(Z^5 + Z^3 + Z^2 + Z\right) z^{32} + z^{29} Z^3 + \left(Z^7 + Z^4 + Z^3 + Z^2 + Z\right) z^{31} + \left(Z^7 + Z^4 + Z^3 + Z\right) z^{30} + \left(Z^7 + Z^5 + Z^4\right) z^{28} + \left(Z^6 + Z^5 + Z^4 + Z\right) z^{27} + \left(Z^7 + Z^5 + Z^4 + 1\right) z^{26} + \left(Z^5 + Z\right) z^{25} + \left(Z^7 + Z^4 + Z^3 + 1\right) z^{24} + \left(Z^2 + Z + 1\right) z^{23} + \left(Z^7 + Z^4 + Z^3 + Z^2 + Z + 1\right) z^{22} + \left(Z^7 + Z^5 + Z^4 + Z + 1\right) z^{21} + \left(Z^7 + Z^6 + Z^4 + 1\right) z^{20} + \left(Z^6 + Z^5 + Z^3 + Z^2 + Z\right) z^{19} + \left(Z + 1\right) z^{18} + \left(Z^7 + Z^6 + Z^3\right) z^{17} + \left(Z^7 + Z^6 + Z^5 + Z\right) z^{16} + \left(Z^7 + Z^6 + Z^3 + Z\right) z^{15} + \left(Z^7 + Z^4 + Z^3 + Z^2 + 1\right) z^{14} + \left(Z^6 + Z^4 + Z^3 + Z^2\right) z^{13} + \left(Z^5 + Z^2 + 1\right) z^{12} + \left(Z^4 + Z^3\right) z^{11} + \left(Z^7 + Z^6 + Z^5 + Z^2 + Z\right) z^{10} + \left(Z^6 + Z^2 + Z\right) z^9 + \left(Z^6 + Z^3 + Z\right) z^8 + \left(Z^7 + Z^5 + Z^4 + 1\right) z^7 + Z^7 + \left(Z^7 + Z^6 + Z^5 + Z^4 + Z^2 + Z\right) z^6 + Z^6 + \left(Z^7 + Z^6 + Z^5 + Z + 1\right) z^5 + Z^5 + \left(Z^7 + Z^4 + Z^2 + Z\right) z^4 + \left(Z^7 + Z^5 + Z^2 + Z + 1\right) z^3 + Z^3 + \left(Z^7 + Z^6 + Z^5 + Z^2\right) z^2 + Z^2 + \left(Z^5 + Z^2 + Z + 1\right) z + 1$$

```
> cv:=CoefficientVector(c,z); cv[1];
  cvb:=map(x->CoefficientList(x,Z),cv): cvb[1];
  cvd:=map(x->add(x[i]*2^(i-1),i=1..nops(x)),cvb): cvd[1];
```

$$cv := \begin{bmatrix} 1 \text{ .. } 255 \; Vector_{column} \\ Data\ Type: anything \\ Storage: rectangular \\ Order: Fortran\_order \end{bmatrix}$$

$$Z^7 + Z^6 + Z^5 + Z^3 + Z^2 + 1$$

$$[1, 0, 1, 1, 0, 1, 1, 1]$$

$$237 \qquad (9.3.25.10)$$

```
> vvd:=cvd; vvd[2]; vvd[5];
  vvd[2]:=cvd[2]+7 mod 2^lgn; vvd[5]:=cvd[5]+13 mod 2^lgn;
```

$$vvd := \begin{bmatrix} 1 .. 255\ Vector_{column} \\ Data\ Type: anything \\ Storage: rectangular \\ Order: Fortran\_order \end{bmatrix}$$

$$39$$

$$150$$

$$vvd_2 := 46$$

$$vvd_5 := 163 \qquad (9.3.25.11)$$

```
> vvb:=map(x->convert(x,base,2),vvd): vvb[2]; vvb[5];
  vv:=map(x->add(x[i]*Z^(i-1),i=1..nops(x)),vvb): vv[2]; vv
  [5];
```

$$[0, 1, 1, 1, 0, 1]$$

$$[1, 1, 0, 0, 0, 1, 0, 1]$$

$$Z^5 + Z^3 + Z^2 + Z$$

$$1 + Z + Z^5 + Z^7 \qquad (9.3.25.12)$$

```
> v:=add(vv[i]*z^(i-1),i=1..n): c:=add(cv[i]*z^(i-1),i=1..n):
  e:=v-c:
  e:=normalizepolyzZ(e,M);
  ev:=Vector(255); ev:=CoefficientVector(e,z);
```

$$e := \left(Z^5 + Z^4 + Z^2 + 1\right) z^4 + \left(Z^3 + 1\right) z$$

$$ev := \begin{bmatrix} 1 .. 255\ Vector_{column} \\ Data\ Type: anything \\ Storage: rectangular \\ Order: Fortran\_order \end{bmatrix}$$

$$ev := \begin{bmatrix} 0 \\ Z^3 + 1 \\ 0 \\ 0 \\ Z^5 + Z^4 + Z^2 + 1 \end{bmatrix} \qquad (9.3.25.13)$$

```
> sindrome:=proc(m,n,alpha,M,Z,sv,vv) local i,j,s,beta,gamma;
  beta:=1;
  for i to m do beta:=modpol(beta*alpha,M,Z,2); gamma:=1; s:=
  0;
    for j from 0 to n-1 do
      s:=modpol(s+vv[j+1]*gamma,M,Z,2); gamma:=modpol(gamma*
  beta,M,Z,2);
```

```
      od; sv[i]:=s;
   od; end;
```

$$sindrome := \mathbf{proc}(m, n, \alpha, M, Z, sv, vv) \tag{9.3.25.14}$$

```
    local i, j, s, β, γ;
    β := 1;
    for i to m do
        β := modpol(β*α, M, Z, 2);
        γ := 1;
        s := 0;
        for j from 0 to n - 1 do
            s := modpol(s + vv[j + 1]*γ, M, Z, 2);
            γ := modpol(γ*β, M, Z, 2)
        end do;
        sv[i] := s
    end do
end proc
```

```
> sv:=Vector(m): sindrome(m,n,alpha,M,Z,sv,vv):
  s:=add(sv[i]*z^(i-1),i=1..m);
```

$$s := Z^6 + Z^4 + Z + 1 + \left(Z^7 + Z^3 + 1\right) z + \left(Z^6 + Z^5 + Z^4 + Z^3 + Z^2 + 1\right) z^2 + \left(Z^5 + Z^4 + Z + 1\right) z^3 + \left(Z^2 + Z^4 + Z^6\right) z^4 + \left(Z^7 + Z^6 + Z^5 + Z^3 + Z^2 + 1\right) z^5 + \left(Z^5 + Z^4 + Z^2 + Z\right) z^6 + \left(Z^7 + Z^6 + Z^5 + Z^4 + Z^3 + Z + 1\right) z^7 + \left(Z^6 + Z^3 + 1\right) z^8 + (Z + 1) z^9 + \left(Z^7 + Z^6 + Z^5 + Z^4 + Z\right) z^{10} + \left(Z^4 + Z + 1\right) z^{11} + \left(Z^7 + Z^6 + Z^4\right) z^{12} + \left(Z^6 + Z^4 + Z^3 + Z^2\right) z^{13} + \left(Z^7 + Z^3 + Z^2 + Z\right) z^{14} + \left(Z^4 + Z^3 + 1\right) z^{15} \tag{9.3.25.15}$$

9.3.26. Reed-Solomon-kód dekódolása.

```
> L:=(1-alpha^1*z)*(1-alpha^4*z); L:=normalizepolyzZ(L,M);
```

$$L := (1 - (Z + 1) z)\left(1 - (Z + 1)^4 z\right)$$

$$L := \left(Z^5 + Z^4 + Z + 1\right) z^2 + \left(Z^4 + Z\right) z + 1 \tag{9.3.26.1}$$

```
> beta:=1: for j from 0 to n-1 do
    subs(z=beta,L); beta:=modpol(beta/alpha,M,Z,2);
  if modpol(%%,M,Z,2)=0 then print(j) fi od:
```

$$1$$

$$4 \tag{9.3.26.2}$$

```
> L1:=(1-alpha^4*z); L4:=(1-alpha^1*z);
```

$$L1 := 1 - (Z + 1)^4 z$$

$$L4 := 1 - (Z + 1) z \tag{9.3.26.3}$$

```
> E:=alpha^1*ev[2]*L1+alpha^4*ev[5]*L4; E:=normalizepolyzZ(E,
  M);
```

$$E := (Z+1)\left(Z^3+1\right)\left(1-(Z+1)^4 z\right)+(Z+1)^4\left(Z^5+Z^4+Z^2+1\right)(1-(Z+1)z)$$

$$E := Z^6+Z^4+\left(Z^6+Z^5+Z^3\right)z+Z+1 \qquad (9.3.26.4)$$

```
> modpol(subs(z=alpha^(-1),E)/alpha^1/subs(z=alpha^(-1),L1),
  M,Z,2);
```

$$Z^3+1 \qquad (9.3.26.5)$$

```
> modpol(subs(z=alpha^(-4),E)/alpha^4/subs(z=alpha^(-4),L4),
  M,Z,2);
```

$$Z^5+Z^4+Z^2+1 \qquad (9.3.26.6)$$

9.3.27. Tétel.

```
> reedsolomondecodezZ:=proc(m::posint,s::polynom,M::polynom)
  local x0,x1,x2,y0,y1,y2,r0,r1,r2,q,i,L,E;
  x0:=1; y0:=0; r0:=z^m; x1:=0; y1:=1; r1:=collect
  (normalizepolyzZ(s,M),z);
  do
    if degree(r1,z)<m/2 then
      L:=CoefficientList(y1,z);
      E:=CoefficientList(r1,z);
      E:=map(x->modpol(x/L[1],M,Z,2),E);
      E:=add(E[i]*z^(i-1),i=1..nops(E));
      L:=map(x->modpol(x/L[1],M,Z,2),L);
      L:=add(L[i]*z^(i-1),i=1..nops(L));
      return [L,E]
    fi;
    r2:=r0; x2:=x0; y2:=y0;
    while degree(r2,z)>=degree(r1,z) do
      q:=modpol(lcoeff(r2,z)/lcoeff(r1,z),M,Z,2);
      q:=q*z^(degree(r2,z)-degree(r1,z));
      r2:=normalizepolyzZ(r2-q*r1,M); r2:=collect(r2,z);
      x2:=normalizepolyzZ(x2-q*x1,M);
      y2:=normalizepolyzZ(y2-q*y1,M);
    od;
    r0:=r1; x0:=x1; y0:=y1; r1:=r2; x1:=x2; y1:=y2;
  od; end;
```

$$reedsolomondecodezZ := \mathbf{proc}(m::posint, s::polynom, M::polynom) \qquad (9.3.27.1)$$

$$\mathbf{local}\ x0, x1, x2, y0, y1, y2, r0, r1, r2, q, i, L, E;$$

$$x0 := 1;$$

$$y0 := 0;$$

$$r0 := z\hat{}m;$$

```
    x1 := 0;
    y1 := 1;
    r1 := collect(normalizepolyzZ(s, M), z);
    do
        if degree(r1, z) < 1 / 2 * m then
            L := PolynomialTools:-CoefficientList(y1, z);
            E := PolynomialTools:-CoefficientList(r1, z);
            E := map(proc(x)
                option operator, arrow;
                modpol(x / L[1], M, Z, 2)
            end proc, E);
            E := add(E[i] * z^(i - 1), i = 1 ..nops(E));
            L := map(proc(x)
                option operator, arrow;
                modpol(x / L[1], M, Z, 2)
            end proc, L);
            L := add(L[i] * z^(i - 1), i = 1 ..nops(L));
            return [L, E]
        end if;
        r2 := r0;
        x2 := x0;
        y2 := y0;
        while degree(r1, z) <= degree(r2, z) do
            q := modpol(lcoeff(r2, z) / lcoeff(r1, z), M, Z, 2);
            q := q * z^(degree(r2, z) - degree(r1, z));
            r2 := normalizepolyzZ(r2 - q * r1, M);
            r2 := collect(r2, z);
            x2 := normalizepolyzZ(x2 - q * x1, M);
            y2 := normalizepolyzZ(y2 - q * y1, M)
        end do;
        r0 := r1;
        x0 := x1;
        y0 := y1;
        r1 := r2;
        x1 := x2;
        y1 := y2
    end do
end proc
```

```
> reedsolomondecodezZ(m,s,M);
```

$$\left[\left(Z^5+Z^4+Z+1\right)z^2+\left(Z^4+Z\right)z+1,\ Z^6+Z^4+\left(Z^6+Z^5+Z^3\right)z+Z+1\right] \quad (9.3.27.2)$$

```
>
```

▼ *9.3.28. Példa.
▼ *9.3.29. Kódrövidítés.
▼ *9.3.30. Kódok direkt szorzata.
▼ *9.3.31. Kaszkád kódok.
▼ *9.3.32. Adatátszövés.
▶ ->9.3.33. Feladat.
▶ 9.3.34. Feladat.
▶ 9.3.35. Feladat.
▼ ->9.3.36. Feladat.
▶ ->9.3.37. Feladat.
▼ ->9.3.38. Feladat.
▼ ->9.3.39. Feladat.
▼ ->9.3.40. Feladat.
▶ ->9.3.41. Feladat.
▼ ->9.3.42. Feladat.
▼ ->9.3.43. Feladat.
▼ ->9.3.44. Feladat.
▼ ->9.3.45. Feladat.
▼ ->9.3.46. Feladat.
▼ *9.3.47. Feladat.
▼ *9.3.48. Feladat.
▶ 9.3.49. Feladat.
▶ 9.3.50. Feladat.
▶ 9.3.51. További feladatok megoldásokkal.
▶ 9.3.52. További feladatok.

▶ 10. Algoritmusok