**Додаток А.**

**Програмний код.**

**unit** Viterbi**; interface**

# uses

Windows**,** Messages**,** SysUtils**,** Variants**,** Classes**,** Graphics**,** Controls**,** Forms**,** Dialogs**,** ExtCtrls**,** StdCtrls**,** Grids**,** ComCtrls**,** Vcl**.**Samples**.**Spin**;**

# type

pint **= ^**integer**;**

int **=** integer**;** TNodeDataPtr **= ^**TNodeData**;**

TNodeData **= record** val**: string;** way**:** int**;**

minWayBelong**:** bool**;** parent**:** TNodeDataPtr**;**

# end;

TListElemPtr **= ^**TListElem**;**

TListElem **= record**

nodes**: array [**0 **..** 3**] of** TNodeDataPtr**;** code**: string;**

pnext**:** TListElemPtr**;** pprev**:** TListElemPtr**;**

# end;

TForm1 **= class(**TForm**)** GroupBox1**:** TGroupBox**;** Button1**:** TButton**;** Button2**:** TButton**;** Button3**:** TButton**;** Button4**:** TButton**;** Label2**:** TLabel**;** Label3**:** TLabel**;** Label4**:** TLabel**;** Label5**:** TLabel**;** Label6**:** TLabel**;** Label7**:** TLabel**;** Label8**:** TLabel**;** RichEdit2**:** TRichEdit**;** RichEdit3**:** TRichEdit**;** RichEdit4**:** TRichEdit**;** RichEdit5**:** TRichEdit**;** RichEdit6**:** TRichEdit**;** RichEdit1**:** TRichEdit**;** ScrollBar1**:** TScrollBar**;** SpinEdit1**:** TSpinEdit**;**

LabeledEdit1**:** TLabeledEdit**;**

**procedure** FormCreate**(**Sender**:** TObject**); procedure** Button1Click**(**Sender**:** TObject**); procedure** Button2Click**(**Sender**:** TObject**); procedure** Button3Click**(**Sender**:** TObject**); procedure** Button4Click**(**Sender**:** TObject**);**

**procedure** FormPaint**(**Sender**:** TObject**);**

**procedure** printTree**(**t**:** TListElemPtr**;** levelNum**:** int**); procedure** step0**();**

**procedure** step1**(); procedure** step2**(); procedure** step3**(); procedure** step4**();**

**procedure** ScrollBar1Change**(**Sender**:** TObject**); private**

bitmap**:** TBitmap**; public**

# end;

MealyAutomaton **= class private**

R1**:** int**;** R2**:** int**;**

# constructor init; public

**procedure** coding**(**X**:** int**;** Y1**:** pint**;** Y2**:** pint**); end;**

TTreeHandler **= class public**

pleft**:** TListElemPtr**;**

**function** xorStrings**(**val\_1**: string;** val\_2**: string): string; constructor** init**;**

**function** calcWayToNode**(**parent**:** TNodeDataPtr**;** nodeVal**: string;**

inputCode**: string):** int**; procedure** creatRoot**;**

**function** createNode**(**parent1**:** TNodeDataPtr**;** parent2**:** TNodeDataPtr**;** nodeVal**: string;** inputCode**: string):** TNodeDataPtr**;**

**function** weightCalc**(**val**: string):** int**;**

**procedure** createLevel**(**temp**:** TListElemPtr**;** code**: string); procedure** deleteTree**(**pleft**:** TListElemPtr**);**

**function** decode**(): string;**

**function** rotateString**(**val**: string): string; end;**

# var

Form1**:** TForm1**;**

mealyCoder**:** MealyAutomaton**;**

treeHandler**:** TTreeHandler**; implementation**

**constructor** MealyAutomaton**.**init**; begin**

R1 **:=** 0**;**

R2 **:=** 0**;**

# end;

**procedure** MealyAutomaton**.**coding**(**X**:** int**;** Y1**:** pint**;** Y2**:** pint**); begin**

Y1**^ :=** X **xor** R2**;** Y2**^ :=** X **xor** R2**;** R2 **:=** R1**;**

R1 **:=** X**;**

Y1**^ :=** Y1**^ xor** R2**; end;**

**function** TTreeHandler**.**xorStrings**(**val\_1**: string;** val\_2**: string): string; var**

res**: string;**

i**:** int**; begin**

**if (**length**(**val\_1**) <>** length**(**val\_2**)) then begin**

res **:=** ''**; end**

# else begin

**for** i **:=** 1 **to** length**(**val\_1**) do begin**

**if (**val\_1**[**i**] =** val\_2**[**i**]) then begin**

res **:=** res **+** '0'

# end else begin

res **:=** res **+** '1'**; end**

# end end;

xorStrings **:=** res**; end;**

**constructor** TTreeHandler**.**init**; begin**

pleft **:= NIL; end;**

**function** TTreeHandler**.**calcWayToNode**(**parent**:** TNodeDataPtr**;** nodeVal**: string;**

inputCode**: string):** int**; var**

resultWay**:** int**; begin**

resultWay **:=** 0**;**

**if (**parent**^.**val **=** '00'**) then begin**

**if (**nodeVal **=** '10'**) then begin**

resultWay **:=** parent**^.**way **+** weightCalc**(**xorStrings**(**'11'**,** inputCode**)); end**

# else begin

resultWay **:=** parent**^.**way **+** weightCalc**(**xorStrings**(**'00'**,** inputCode**)); end**

# end

**else if (**parent**^.**val **=** '10'**) then begin**

**if (**nodeVal **=** '11'**) then begin**

resultWay **:=** parent**^.**way **+** weightCalc**(**xorStrings**(**'01'**,** inputCode**)); end**

# else begin

resultWay **:=** parent**^.**way **+** weightCalc**(**xorStrings**(**'10'**,** inputCode**)); end**

# end

**else if (**parent**.**val **=** '11'**) then begin**

**if (**nodeVal **=** '11'**) then begin**

resultWay **:=** parent**^.**way **+** weightCalc**(**xorStrings**(**'10'**,** inputCode**)); end**

# else begin

resultWay **:=** parent**^.**way **+** weightCalc**(**xorStrings**(**'01'**,** inputCode**)); end**

# end

**else if (**parent**^.**val **=** '01'**) then begin**

**if (**nodeVal **=** '10'**) then begin**

resultWay **:=** parent**^.**way **+** weightCalc**(**xorStrings**(**'00'**,** inputCode**)); end**

# else begin

resultWay **:=** parent**^.**way **+** weightCalc**(**xorStrings**(**'11'**,** inputCode**)); end**

# end;

calcWayToNode **:=** resultWay**; end;**

**procedure** TTreeHandler**.**creatRoot**; begin**

new**(**pleft**);** pleft**^.**pnext **:= NIL;** pleft**^.**pprev **:= NIL;** new**(**pleft**^.**nodes**[**0**]);**

pleft**^.**nodes**[**0**]^.**val **:=** '00'**;**

pleft**^.**nodes**[**0**]^.**way **:=** 0**;** pleft**^.**nodes**[**0**]^.**minWayBelong **:=** true**;** pleft**^.**nodes**[**0**]^.**parent **:= NIL;** pleft**^.**nodes**[**1**] := NIL;** pleft**^.**nodes**[**2**] := NIL;** pleft**^.**nodes**[**3**] := NIL;**

# end;

**function** TTreeHandler**.**createNode**(**parent1**:** TNodeDataPtr**;** parent2**:** TNodeDataPtr**;** nodeVal**: string;** inputCode**: string):** TNodeDataPtr**;**

# var

nodeParent**:** TNodeDataPtr**;** resultNode**:** TNodeDataPtr**;** way1**:** int**;**

way2**:** int**;** resWay**:** int**;**

# begin

**if ((**parent1 **= NIL) AND (**parent2 **= NIL)) then begin**

resultNode **:= NIL; end**

# else begin

**if (**parent1 **= NIL) then begin**

nodeParent **:=** parent2**; end**

**else if (**parent2 **= NIL) then begin**

nodeParent **:=** parent1**; end**

# else begin

way1 **:=** calcWayToNode**(**parent1**,** nodeVal**,** inputCode**);** way2 **:=** calcWayToNode**(**parent2**,** nodeVal**,** inputCode**); if (**way2 **>** way1**) then**

nodeParent **:=** parent1

# else

nodeParent **:=** parent2**; end;**

resWay **:=** calcWayToNode**(**nodeParent**,** nodeVal**,** inputCode**);** new**(**resultNode**);**

resultNode**^.**val **:=** nodeVal**;** resultNode**^.**way **:=** resWay**;** resultNode**^.**minWayBelong **:=** false**;**

resultNode**^.**parent **:=** nodeParent**; end;**

createNode **:=** resultNode**; end;**

**function** TTreeHandler**.**weightCalc**(**val**: string):** int**; var**

res**:** int**;** i**:** int**;**

# begin

res **:=** 0**;**

**for** i **:=** 1 **to** length**(**val**) do begin**

**if (**val**[**i**] =** '1'**) then begin**

res **:=** res **+** 1**; end**

# end;

weightCalc **:=** res**; end;**

**procedure** TTreeHandler**.**createLevel**(**temp**:** TListElemPtr**;** code**: string); var**

newLevel**:** TListElemPtr**; begin**

**if ((**temp**^.**pnext **<> NIL)) then begin**

createLevel**(**temp**.**pnext**,** code**); end**

# else begin

new**(**newLevel**);** temp**^.**pnext **:=** newLevel**;** temp**^.**code **:=** code**;** newLevel**^.**pnext **:= NIL;** newLevel**^.**pprev **:=** temp**;**

newLevel**^.**nodes**[**0**] :=** createNode**(**temp**^.**nodes**[**0**],** temp**^.**nodes**[**2**],** '00'**,** code**);**

newLevel**^.**nodes**[**1**] :=** createNode**(**temp**^.**nodes**[**0**],** temp**^.**nodes**[**2**],** '10'**,** code**);**

newLevel**^.**nodes**[**2**] :=** createNode**(**temp**^.**nodes**[**1**],** temp**^.**nodes**[**3**],** '01'**,** code**);**

newLevel**^.**nodes**[**3**] :=** createNode**(**temp**^.**nodes**[**1**],** temp**^.**nodes**[**3**],** '11'**,** code**);**

# end end;

**procedure** TTreeHandler**.**deleteTree**(**pleft**:** TListElemPtr**); begin**

**if (**pleft **<> NIL) then begin**

deleteTree**(**pleft**.**pnext**);** DISPOSE**(**pleft**);**

# end;

pleft **:= NIL;**

# end;

**function** TTreeHandler**.**decode**(): string; var**

temp**:** TListElemPtr**;** resultStr**: string;** minWayNode**:** TNodeDataPtr**;** i**:** int**;**

# begin

temp **:=** pleft**;**

**while (**temp**^.**pnext **<> NIL) do begin**

temp **:=** temp**^.**pnext**; end;**

minWayNode **:= NIL; for** i **:=** 0 **to** 3 **do begin**

**if (**temp**^.**nodes**[**i**] = NIL) then begin**

break**; end;**

**if ((**minWayNode **= NIL) OR (**minWayNode**^.**way **>** temp**^.**nodes**[**i**]^.**way**)) then begin**

minWayNode **:=** temp**^.**nodes**[**i**]; end;**

# end;

**while (**minWayNode**^.**parent **<> NIL) do begin**

resultStr **:=** resultStr **+** minWayNode**^.**val**[**1**];** minWayNode**^.**minWayBelong **:=** true**;** minWayNode **:=** minWayNode**^.**parent**;**

# end;

decode **:=** rotateString**(**resultStr**); end;**

**function** TTreeHandler**.**rotateString**(**val**: string): string; var**

res**: string;** strLen**:** int**;** i**:** int**;**

# begin

res **:=** ''**;**

strLen **:=** length**(**val**);** i **:=** strLen**;**

# while (i >= 1) do begin

res **:=** res **+** val**[**i**];** i **:=** i **-** 1**;**

# end;

rotateString **:=** res**; end;**

**procedure** TForm1**.**FormCreate**(**Sender**:** TObject**); begin**

mealyCoder **:=** MealyAutomaton**.**init**();**

treeHandler **:=** TTreeHandler**.**init**();** Form1**.**DoubleBuffered **:=** true**;** bitmap **:=** TBitmap**.**Create**;** randomize**();**

RichEdit1**.**Clear**();** RichEdit1**.**Text **:=** 'тест1.'**;**

# end;

**procedure** TForm1**.**printTree**(**t**:** TListElemPtr**;** levelNum**:** int**); var**

i**:** int**;**

node**:** TNodeDataPtr**;** nodeParent**:** TNodeDataPtr**;** leftPos**:** int**;**

topPos**:** int**;** levelHeight**:** int**;** levelWidth**:** int**;** levelX**:** int**;** levelY**:** int**;** ellipse**:** int**;** parentNodeY**:** int**;** parentNodeX**:** int**;** max**:** int**;**

# begin

leftPos **:=** 60 **-** ScrollBar1**.**Position**;** topPos **:=** 40 **;**

levelHeight **:=** 35**;**

levelWidth **:=** 60**;**

ellipse **:=** 12**;**

parentNodeX **:=** 0**;**

parentNodeY **:=** 0**;**

bitmap**.**Width **:=** Form1**.**Width**;** bitmap**.**Height **:=** 217**;** bitmap**.**Canvas**.**Brush**.**Color **:=** clBtnFace**;** bitmap**.**Canvas**.**Pen**.**Style **:=** psSolid**;** bitmap**.**Canvas**.**Pen**.**Color **:=** clBlack**;**

bitmap**.**Canvas**.**FillRect**(**Rect**(**0**,** 0**,** Form1**.**Width**,** Form1**.**Height**));**

# while (t <> NIL) do begin

levelX **:=** leftPos **+** levelWidth **\*** levelNum**; for** i **:=** 0 **to** 3 **do**

# begin

**if (**t**^.**nodes**[**i**] <> NIL) then begin**

levelY **:=** topPos **+** levelHeight **\*** i**;** node **:=** t**^.**nodes**[**i**];**

nodeParent **:=** node**^.**parent**; if (**nodeParent **<> NIL) then begin**

**if (**nodeParent**.**val **=** '00'**) then**

parentNodeY **:=** 0

**else if (**nodeParent**.**val **=** '10'**) then**

parentNodeY **:=** 1

**else if (**nodeParent**.**val **=** '01'**) then**

parentNodeY **:=** 2

# else

parentNodeY **:=** 3**;**

parentNodeY **:=** topPos **+** levelHeight **\*** parentNodeY**;** parentNodeX **:=** levelX **-** levelWidth**;**

**if (**node**^.**minWayBelong**) then begin**

bitmap**.**Canvas**.**Pen**.**Color **:=** clRed**;** bitmap**.**Canvas**.**Pen**.**Width **:=** 2**;**

# end else begin

bitmap**.**Canvas**.**Pen**.**Color **:=** clBlack**;** bitmap**.**Canvas**.**Pen**.**Width **:=** 1**;**

# end;

bitmap**.**Canvas**.**MoveTo**(**levelX**,** levelY**);** bitmap**.**Canvas**.**LineTo**(**parentNodeX**,** parentNodeY**);**

# end;

bitmap**.**Canvas**.**Pen**.**Color **:=** clBlack**;** bitmap**.**Canvas**.**Pen**.**Width **:=** 1**;**

bitmap**.**Canvas**.**ellipse**(**levelX **-** ellipse**,** levelY **-** ellipse**,** levelX **+** ellipse**,** levelY **+** ellipse**);**

bitmap**.**Canvas**.**TextOut**(**levelX **-** 7**,** levelY **-** 6**,** IntToStr**(**node**^.**way**));**

**if (**nodeParent **<> NIL) then begin**

bitmap**.**Canvas**.**ellipse**(**parentNodeX **-** ellipse**,** parentNodeY **-** ellipse**,** parentNodeX **+** ellipse**,** parentNodeY **+** ellipse**);**

bitmap**.**Canvas**.**TextOut**(**parentNodeX **-** 7**,** parentNodeY **-** 6**,** IntToStr**(**nodeParent**^.**way**));**

# end; end;

bitmap**.**Canvas**.**TextOut**(**levelX **-** 7**,** topPos **-** 6 **+** levelHeight **\*** 4**,** IntToStr**(**levelNum **+** 1**));**

bitmap**.**Canvas**.**TextOut**(**levelX **-** 7 **+** trunc**(**levelWidth **/** 2**),** 10**,** t**^.**code**); end;**

t **:=** t**^.**pnext**;**

levelNum **:=** levelNum **+** 1**; end;**

**if (**treeHandler**.**pleft **<> NIL) then begin**

bitmap**.**Canvas**.**FillRect**(**Rect**(**0**,** 0**,** 40**,** Form1**.**Height**));**

bitmap**.**Canvas**.**TextOut**(**10**,** topPos **-** 6**,** '00'**);** bitmap**.**Canvas**.**TextOut**(**10**,** topPos **-** 6 **+** levelHeight**,** '10'**);** bitmap**.**Canvas**.**TextOut**(**10**,** topPos **-** 6 **+** levelHeight **\*** 2**,** '01'**);**

bitmap**.**Canvas**.**TextOut**(**10**,** topPos **-** 6 **+** levelHeight **\*** 3**,** '11'**);**

Form1**.**Canvas**.**Draw**(**0**,** 0**,** bitmap**);**

max **:=** levelNum **\*** levelWidth **-** Form1**.**Width **+** 40**; if (**max **>** 0**) then**

# begin

ScrollBar1**.**Visible **:=** true**;** ScrollBar1**.**max **:=** max**;**

# end else begin

ScrollBar1**.**Visible **:=** false**; end**

# end else begin

ScrollBar1**.**Visible **:=** false**; end;**

# end;

**procedure** TForm1**.**step0**(); var**

sym**:** char**;** i**:** int**;**

j**:** int**;** str**: string;**

# begin

RichEdit2**.**Text **:=** ''**;**

**for** i **:=** 1 **to** length**(**RichEdit1**.**Text**) do begin**

sym **:=** RichEdit1**.**Text**[**i**];** str **:=** ''**;**

**for** j **:=** 0 **to** 7 **do begin**

**if (**int**(**sym**) and (**1 **shl** j**) >** 0**) then**

str **:=** '1' **+** str

# else

str **:=** '0' **+** str**; end;**

RichEdit2**.**Text **:=** RichEdit2**.**Text **+** str**; end;**

Label2**.**Caption **:=** 'Повідомлення в двійковому коді (' **+**

IntToStr**(**length**(**RichEdit2**.**Text**)) +** ' біт)'**; end;**

**procedure** TForm1**.**step1**(); var**

Y1**:** int**;** Y2**:** int**;** res**: string;** i**:** int**;** sym**:** int**;**

# begin

res **:=** ''**;** mealyCoder**.**init**();**

**for** i **:=** 1 **to** length**(**RichEdit2**.**Text**) do begin**

**if ((**RichEdit2**.**Text**[**i**] =** '0'**) OR (**RichEdit2**.**Text**[**i**] =** '1'**)) then begin**

sym **:=** int**(**RichEdit2**.**Text**[**i**]) -** int**(**'0'**);** mealyCoder**.**coding**(**sym**, @**Y1**, @**Y2**);**

res **:=** res **+** IntToStr**(**Y1**) +** IntToStr**(**Y2**);**

# end; end;

RichEdit3**.**Text **:=** res**;**

Label3**.**Caption **:=** 'Закодоване повідомлення (' **+**

IntToStr**(**length**(**RichEdit3**.**Text**)) +** ' біт)'**; end;**

**procedure** TForm1**.**step2**(); var**

i**:** int**;**

j**:** int**;** errorMultiplicity**:** int**;** errorProbability**:** int**;** errorInterval**:** int**;** errorPosition**:** int**;** str**: string;** maxErrors**:** int**;** errorCount**:** int**;**

# begin

errorInterval **:=** 1**;** errorMultiplicity **:=** SpinEdit1**.**Value**;**

errorProbability **:=** trunc**(**StrToFloat**(**LabeledEdit1**.**Text**) \*** 100**);**

RichEdit4**.**SelStart **:=** 1**;**

RichEdit4**.**SelLength **:=** length**(**RichEdit4**.**Text**);** RichEdit4**.**SelAttributes**.**Color **:=** clBlack**;** RichEdit4**.**SelAttributes**.**Style **:= [];** RichEdit4**.**Lines**.**Clear**();**

str **:=** RichEdit3**.**Text**;** randomize**();**

maxErrors **:=** round**(**length**(**str**) \* (**errorProbability **/** 100**));** errorCount **:=** 0**;**

i **:=** 1**;**

**while ((**i **<=** length**(**str**)) AND (**maxErrors **>** 0**)) do begin**

**if (**errorCount **>=** maxErrors**) then begin**

break**; end;**

**if (**random**(**99**) +** 1 **<=** errorProbability**) then begin**

errorPosition **:=** 0**;**

**for** j **:=** i **+** errorMultiplicity **to** i **+** errorPosition **+**

errorMultiplicity **-** 1 **do begin**

**if (**j **>** length**(**str**)) then begin**

break**; end;**

**if (**str**[**j**] =** '0'**) then begin**

str**[**j**] :=** '1'

# end else begin

str**[**j**] :=** '0'**; end;**

errorCount **:=** errorCount **+** 1**;** i **:=** i **+** 1**;**

# end;

i **:=** i **-** 1**;**

**if ((**errorProbability **<>** 100**) AND (**random**(**99**) +** 1 **>** errorProbability**)) then**

# begin

i **:=** i **+** 1**;**

# end; end;

i **:=** i **+** errorInterval**; end;**

RichEdit4**.**Text **:=** str**;**

Label6**.**Caption **:=** 'Після проходження каналу (' **+**

IntToStr**(**length**(**RichEdit4**.**Text**)) +** ' біт)'**;**

j**:=** length**(**str**);**

**for** i **:=** 1 **to** length**(**str**) do begin**

**if (**str**[**i**] <>** RichEdit3**.**Text**[**i**]) then begin**

RichEdit4**.**SelStart **:=** i **-** 1**;** RichEdit4**.**SelLength **:=** 1**;** RichEdit4**.**SelAttributes**.**Color **:=** clFuchsia**;**

RichEdit4**.**SelAttributes**.**Style **:= [**fsUnderline**,** fsBold**]; end;**

# end end;

**procedure** TForm1**.**step3**(); var**

code**: string;** codeLen**:** int**;** i**:** int**;**

# begin

code **:=** RichEdit4**.**Text**;** codeLen **:=** length**(**code**); if (**codeLen **>** 0**) then begin**

treeHandler**.**deleteTree**(**treeHandler**.**pleft**);** treeHandler**.**creatRoot**();**

i **:=** 1**;**

**while (**i **<** codeLen**) do begin**

treeHandler**.**createLevel**(**treeHandler**.**pleft**,** copy**(**code**,** i**,** 2**));** i **:=** i **+** 2**;**

# end;

RichEdit5**.**Text **:=** treeHandler**.**decode**();**

printTree**(**treeHandler**.**pleft**,** 0**); end;**

# end;

**procedure** TForm1**.**step4**(); var**

str**: string;**

i**:** int**;**

j**:** int**;** sym**:** int**;**

# begin

str **:=** ''**;** i **:=** 1**;**

**while (**i **<** length**(**RichEdit5**.**Text**)) do begin**

sym **:=** 0**;**

**for** j **:=** 0 **to** 7 **do begin**

sym **:= (**sym **shl** 1**) + (**byte**(**RichEdit5**.**Text**[**i **+** j**]) and** 1**); end;**

str **:=** str **+** char**(**sym**);** i **:=** i **+** 8**;**

# end;

RichEdit6**.**Text **:=** str**; end;**

**procedure** TForm1**.**Button1Click**(**Sender**:** TObject**); begin**

step0**();**

step1**(); end;**

**procedure** TForm1**.**Button2Click**(**Sender**:** TObject**); begin**

step2**(); end;**

**procedure** TForm1**.**Button3Click**(**Sender**:** TObject**); begin**

step3**();**

step4**(); end;**

**procedure** TForm1**.**Button4Click**(**Sender**:** TObject**); begin**

step0**();**

step1**();**

step2**();**

step3**();**

step4**(); end;**

**procedure** TForm1**.**ScrollBar1Change**(**Sender**:** TObject**); begin**

printTree**(**treeHandler**.**pleft**,** 0**); end;**

**procedure** TForm1**.**FormPaint**(**Sender**:** TObject**); begin**

printTree**(**treeHandler**.**pleft**,** 0**); end;**

# end.