Національний технічний університет України

«Київський політехнічний інститут»

Факультет інформатики та обчислювальної техніки

Кафедра обчислювальної техніки

**Лабораторна робота №1.**

**Множини. Основні властивості та операції над ними**

Виконав: студент І курсу ФІОТ гр. ІО-41 Логвинчук А. І. Перевірив: Новотарський М. К.

Київ

2015

**Множини. Основні властивості та операції над ними**

**Мета:** вивчити основні аксіоми, закони і теореми теорії множин, навчитися

застосовувати їх на практиці. Виконати наступні операції над множинами:

доповнення множин; об’єднання, перетин, різниця, симетрична різниця

**Завдання:** написати програму для виконання даних операцій над множинами.

Варіант: **4120** mod **10** = **0**

**Теоритичні відомості:**

***Множина*** - є сукупність визначених об'єктів, різних між собою, об’єднаних за певною ознакою чи властивістю.

***Операції над множинами***

***Об’єднання.*** Об'єднанням множин A і B називається множина, що складається

із всіх тих елементів, які належать хоча б однієї з множин A або B . Об'єднання

множин A і B позначається A U B. Це визначення рівносильне наступному:

A U B = {x | x ∈ A або x ∈ B}.

***Перетин.*** Перетином множин A і B називається множина, що складається із

всіх тих елементів, які належать і множині A й множині B . Перетин множин A

і B позначається A ∩ B. Це визначення рівносильне наступному:

A ∩ B = {x | x ∈ A і x ∈ B}.

***Доповнення.*** Доповненням (або абсолютним доповненням) множини A

називається множина, що складається із всіх елементів універсальної множини,

які не належать A. Доповнення множини A позначається . Це визначення

рівносильне наступному: = U - A.

***Різницею*** множин A і B (або відносним доповненням) називається

множина, що складається із всіх елементів множини A, які не належать B

Різницю множин A і B позначають A \ B.

**Блок-схеми алгоритмів операцій та опис програми**

****

Рис. 1 Об’єднання двох множин



Рис. 2 Перетин двох множин



Рис. 3 Різниця двох множин

В моїй програмі множини в пам’яті ЕОМ представлені у вигляді одновимірного масиву. Всі операції здійснюються над масивами, а результат для наочності виводиться у прямокутну таблицю.

Операція доповнення множини до універсальної аналогічна операції віднімання.

Симетричну різницю знаходимо шляхом послідовного виконання двох операцій віднімання та об’єднання отриманих множин: R1=A\B, R2=B\A, C=R1 U R2.

**Код програми**

unit Unit12;

{$mode objfpc}{$H+}

interface

uses

Classes, SysUtils, FileUtil, Forms, Controls, Graphics, Dialogs, StdCtrls,

ExtCtrls, Grids;

type

TIndex = 0..120;

TArray = Array[TIndex] of string;

{ TOperForm }

TOperForm = class(TForm)

BtoU: TButton;

BminusA: TButton;

AtoU: TButton;

Label1: TLabel;

Label2: TLabel;

Label3: TLabel;

Label4: TLabel;

LoadSetA: TButton;

SetSymmetricDifference: TButton;

SetUnion: TButton;

LoadSetB: TButton;

ExitBtn: TButton;

InfoPanel: TPanel;

SetClear: TButton;

SetA: TStringGrid;

SetB: TStringGrid;

SetC: TStringGrid;

SetIntersection: TButton;

AminusB: TButton;

procedure AminusBClick(Sender: TObject);

procedure AtoUClick(Sender: TObject);

procedure BminusAClick(Sender: TObject);

procedure BtoUClick(Sender: TObject);

function GetLength(P: TArray): TIndex;

procedure LoadSetAClick(Sender: TObject);

procedure LoadSetBClick(Sender: TObject);

procedure ExitBtnClick(Sender: TObject);

procedure SetClearClick(Sender: TObject);

procedure SetIntersectionClick(Sender: TObject);

procedure SetSymmetricDifferenceClick(Sender: TObject);

procedure SetUnionClick(Sender: TObject);

private

{ private declarations }

public

{ public declarations }

end;

var

OperForm: TOperForm;

A,B,C: TArray;

implementation

{$R \*.lfm}

function TOperForm.GetLength(P: TArray): TIndex;

var

i: TIndex;

begin

for i:=0 to 120 do

if P[i]='' then break;

GetLength:=i

end;

procedure TOperForm.AminusBClick(Sender: TObject);

var

i,j,k: TIndex;

contains: boolean;

begin

k:=0;

for i:=0 to GetLength(A)-1 do

begin

contains:=false;

for j:=0 to GetLength(B)-1 do

if A[i]=B[j] then

contains:=true;

if not contains then

begin

C[k]:=A[i];

inc(k)

end;

end;

k:=0;

for i:=0 to 10 do

for j:=0 to 10 do

begin

SetC.Cells[j,i]:=C[k];

k:=k+1;

end;

end;

procedure TOperForm.AtoUClick(Sender: TObject);

var

j,k: TIndex;

i: integer;

U: TArray;

contains: boolean;

begin

k:=0;

for i:=30 to 130 do

begin

U[k]:=Chr(i);

inc(k);

end;

k:=0;

for i:=0 to GetLength(U)-1 do

begin

contains:=false;

for j:=0 to GetLength(A)-1 do

if U[i]=A[j] then

contains:=true;

if not contains then

begin

C[k]:=U[i];

inc(k)

end;

end;

k:=0;

for i:=0 to 10 do

for j:=0 to 10 do

begin

SetC.Cells[j,i]:=C[k];

inc(k);

end;

end;

procedure TOperForm.BminusAClick(Sender: TObject);

var

i,j,k: TIndex;

contains: boolean;

begin

k:=0;

for i:=0 to GetLength(B)-1 do

begin

contains:=false;

for j:=0 to GetLength(A)-1 do

if B[i]=A[j] then

contains:=true;

if not contains then

begin

C[k]:=B[i];

inc(k)

end;

end;

k:=0;

for i:=0 to 10 do

for j:=0 to 10 do

begin

SetC.Cells[j,i]:=C[k];

k:=k+1;

end;

end;

procedure TOperForm.BtoUClick(Sender: TObject);

var

j,k: TIndex;

i: integer;

U: TArray;

contains: boolean;

begin

k:=0;

for i:=30 to 130 do

begin

U[k]:=Chr(i);

inc(k);

end;

k:=0;

for i:=0 to GetLength(U)-1 do

begin

contains:=false;

for j:=0 to GetLength(B)-1 do

if U[i]=B[j] then

contains:=true;

if not contains then

begin

C[k]:=U[i];

inc(k)

end;

end;

k:=0;

for i:=0 to 10 do

for j:=0 to 10 do

begin

SetC.Cells[j,i]:=C[k];

inc(k);

end;

end;

procedure TOperForm.LoadSetAClick(Sender: TObject);

var

i,j,k: Tindex;

MyFile: TextFile;

begin

k:=0;

AssignFile(MyFile, 'DATA\A.txt');

Reset(MyFile);

for i:=0 to 120 do

if not eof(MyFile) then

Readln(MyFile, A[i]);

CloseFile(MyFile);

for i:=0 to 10 do

for j:=0 to 10 do

begin

SetA.Cells[j,i]:=A[k];

inc(k);

end;

end;

procedure TOperForm.LoadSetBClick(Sender: TObject);

var

i,j,k: Tindex;

MyFile: TextFile;

begin

k:=0;

AssignFile(MyFile, 'DATA\B.txt');

Reset(MyFile);

for i:=0 to 120 do

if not eof(MyFile) then

Readln(MyFile, B[i]);

CloseFile(MyFile);

for i:=0 to 10 do

for j:=0 to 10 do

begin

SetB.Cells[j,i]:=B[k];

k:=k+1;

end;

end;

{ TOperForm }

procedure TOperForm.ExitBtnClick(Sender: TObject);

begin

Close;

end;

procedure TOperForm.SetClearClick(Sender: TObject);

var

i,j,k: TIndex;

begin

k:=GetLength(A)-1;

for i:=0 to k do

A[i]:='';

k:=GetLength(B)-1;

for i:=0 to k do

B[i]:='';

k:=GetLength(C)-1;

for i:=0 to k do

C[i]:='';

k:=0;

for i:=0 to 10 do

for j:=0 to 10 do

begin

SetA.Cells[j,i]:='';

inc(k)

end;

k:=0;

for i:=0 to 10 do

for j:=0 to 10 do

begin

SetB.Cells[j,i]:='';

inc(k)

end;

k:=0;

for i:=0 to 10 do

for j:=0 to 10 do

begin

SetC.Cells[j,i]:='';

inc(k)

end;

end;

procedure TOperForm.SetIntersectionClick(Sender: TObject);

var

i,j,k: integer;

begin

k:=0;

for i:=0 to GetLength(B)-1 do

for j:=0 to GetLength(A)-1 do

if B[i]=A[j] then

begin

C[k]:=B[i];

inc(k)

end;

k:=0;

for i:=0 to 10 do

for j:=0 to 10 do

begin

SetC.Cells[j,i]:=C[k];

inc(k);

end;

end;

procedure TOperForm.SetSymmetricDifferenceClick(Sender: TObject);

var

i,j,k: TIndex;

R1,R2: TArray;

contains: boolean;

begin

k:=0;

for i:=0 to GetLength(B)-1 do

begin

contains:=false;

for j:=0 to GetLength(A)-1 do

if B[i]=A[j] then

contains:=true;

if not contains then

begin

R1[k]:=B[i];

inc(k)

end;

end;

k:=0;

for i:=0 to GetLength(A)-1 do

begin

contains:=false;

for j:=0 to GetLength(B)-1 do

if A[i]=B[j] then

contains:=true;

if not contains then

begin

R2[k]:=A[i];

inc(k)

end;

end;

for i:=0 to GetLength(R1)-1 do

C[i]:=R1[i];

k:=GetLength(R1);

for i:=0 to GetLength(R2)-1 do

begin

contains:=false;

for j:=0 to GetLength(R1)-1 do

if R2[i]=R1[j] then

contains:=true;

if not contains then

begin

C[k]:=R2[i];

inc(k);

end;

end;

k:=0;

for i:=0 to 10 do

for j:=0 to 10 do

begin

SetC.Cells[j,i]:=C[k];

inc(k);

end;

end;

procedure TOperForm.SetUnionClick(Sender: TObject);

var

i,j,k: TIndex;

contains: boolean;

begin

for i:=0 to GetLength(A)-1 do

C[i]:=A[i];

k:=GetLength(A);

for i:=0 to GetLength(B)-1 do

begin

contains:=false;

for j:=0 to GetLength(A)-1 do

if B[i]=A[j] then

contains:=true;

if not contains then

begin

C[k]:=B[i];

inc(k);

end;

end;

k:=0;

for i:=0 to 10 do

for j:=0 to 10 do

begin

SetC.Cells[j,i]:=C[k];

inc(k);

end;

end;

end.

**Приклади роботи програми**

Для кожної з наступних операцій програма створює відповідний вивід.

*Об’єднання:*



*Перетин:*

**

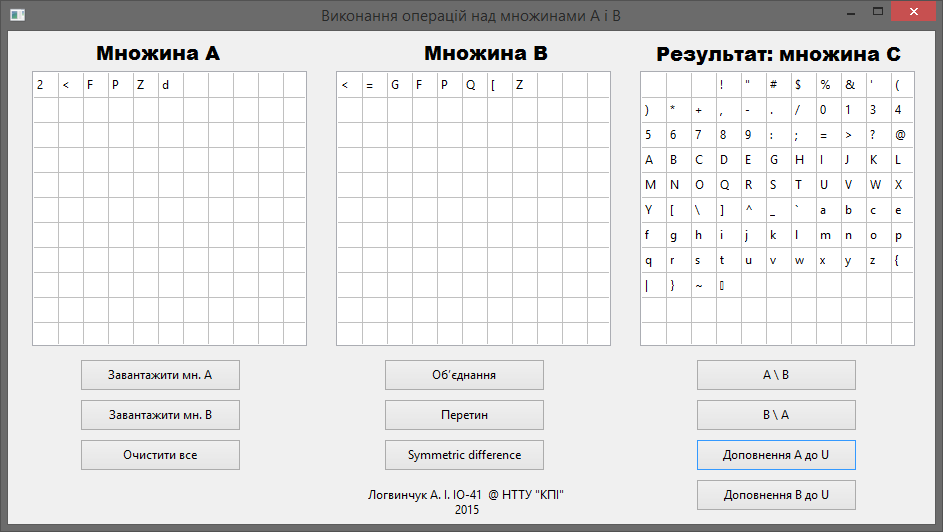
*Симетрична різниця:*

**

*Різниця В\А:*

**

*Доповнення А до U:*



**Висновок:**

Виконуючи роботу я закріпив знання аксіом, теорем і законів теорії множин, навчився застосовувати їх на практиці, реалізував алгоритми основних операцій над множинами (об’єднання, перетин, різниця, доповнення, симетрична різниця) на мові програмування Object Pascal.