Министерство науки и высшего образования РФ

Федеральное государственное бюджетное образовательное учреждение

высшего образования

«Новгородский государственный университет имени Ярослава Мудрого»

Кафедра информационных технологий и систем

**ИСПОЛЬЗОВАНИЕ ОТСЕЧЕНИЙ**

Лабораторная работа по дисциплине:

“Функциональное и логическое программирование”

Выполнил студент группы 8091:

\_\_\_\_\_\_\_\_\_\_ / Васильев И.В.

«\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2021 г.

Принял преподаватель:

\_\_\_\_\_\_ / Михайлов Д.В.

«\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2021 г.

**Великий Новгород**

**2021**

# Цель и задачи

Целью работы является изучение правил использования отсечения в двух случаях: для подтверждения правильности выбранного решения и для прекращения процесса порождения и проверки возможных решений.

Задача №1  
Создать меню, обеспечить вызов и выполнение всех заданий из Лабораторной работы № 6. Каждое задание должно выполняться в отдельном окне. При выполнении работы в среде Visual Prolog (независимо от версии) рекомендуется использовать средства создания графического интерфейса (см. ниже, а также прилагаемые примеры из предыдущей лабораторной работы). Реализовать процедуру ввода списка. При использовании среды визуальной разработки Visual Prolog изучить принципы расстановки отсечений в предложениях, которые генерируются экспертом кода и обрабатывают события от пунктов меню.

Задача №2  
Написать для игры «Крестики-нолики» процедуру «Следующий ход», которая для заданного положения на доске находила бы наилучший ход, обеспечивающий либо предотвращение проигрыша, либо выигрыш, либо наилучший прогнозируемый результат.

Задача №3

Написать программу, реализующую выигрышную стратегию для игры «Крестики-нолики»

на доске 3 × 3.

# Решение

# Задача №1

Создать меню, обеспечить вызов и выполнение всех заданий из Лабораторной работы № 6. Каждое задание должно выполняться в отдельном окне. При выполнении работы в среде Visual Prolog (независимо от версии) рекомендуется использовать средства создания графического интерфейса (см. ниже, а также прилагаемые примеры из предыдущей лабораторной работы). Реализовать процедуру ввода списка. При использовании среды визуальной разработки Visual Prolog изучить принципы расстановки отсечений в предложениях, которые генерируются экспертом кода и обрабатывают события от пунктов меню.

Программный код:

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Copyright (c) My Company

Project: LAB7

FileName: LAB7.PRO

Purpose: No description

Written by: Visual Prolog

Comments:

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

include "Lab7.inc"

include "Lab7.con"

include "hlptopic.con"

include "lab61.pro"

include "lab62.pro"

include "lab63.pro"

include "lab64.pro"

include "xo.pro"

%BEGIN\_WIN Task Window

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Event handling for Task Window

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

predicates

task\_win\_eh : EHANDLER

constants

%BEGIN Task Window, CreateParms, 16:06:53-27.1.2021, Code automatically updated!

task\_win\_Flags = [wsf\_SizeBorder,wsf\_TitleBar,wsf\_Close,wsf\_Maximize,wsf\_Minimize,wsf\_ClipSiblings]

task\_win\_Menu = res\_menu(idr\_task\_menu)

task\_win\_Title = "Lab7"

task\_win\_Help = idh\_contents

%END Task Window, CreateParms

clauses

% Shell sorting

task\_win\_eh(\_Win,e\_Menu(id\_ex\_1,\_ShiftCtlAlt),0):-

!,

S=dlg\_GetStr("Enter list","Enter list. Example: [1, 2, 3, 4, 5]",""),

term\_str(lst,Lst,S),

sort(Lst, Res),

term\_str(lst,Res,S1),

dlg\_MessageBox(

"Result",

S1,

mesbox\_iconInformation,

mesbox\_buttonsOK,

mesbox\_defaultFirst,

mesbox\_suspendApplication

),

!.

% Insert Sorting

task\_win\_eh(\_Win,e\_Menu(id\_ex\_2,\_ShiftCtlAlt),0):-

!,

S=dlg\_GetStr("Enter list","Enter list. Example: [1, 2, 3, 4, 5]",""),

term\_str(lst1,Lst,S),

selectionsort(Lst, Res),

term\_str(lst1,Res,S1),

dlg\_MessageBox(

"Result",

S1,

mesbox\_iconInformation,

mesbox\_buttonsOK,

mesbox\_defaultFirst,

mesbox\_suspendApplication

),

!.

% Insert sublist in list from N position

task\_win\_eh(\_Win,e\_Menu(id\_ex\_3,\_ShiftCtlAlt),0):-

!,

S=dlg\_GetStr("Enter list","Enter FIRST list. Example: [1, 2, 3, 4, 5]",""),

term\_str(listz,List1,S),

S2=dlg\_GetStr("Enter SUBList","Enter SUBlist. Example: [6, 7]",""),

term\_str(listz,List2,S2),

S3=dlg\_GetStr("Enter position","Enter position N = . Example: 3",""),

term\_str(intz,N,S3),

mainz(List1,List2,N, Final),

term\_str(listz,Final,S4),

dlg\_MessageBox(

"Result",

S4,

mesbox\_iconInformation,

mesbox\_buttonsOK,

mesbox\_defaultFirst,

mesbox\_suspendApplication

),

!.

% Game

task\_win\_eh(\_Win,e\_Menu(id\_ex\_4,\_ShiftCtlAlt),0):-

!,

game(

[0,0,0,1,0,2,0,3,0,4,0,5,0,6,

1,1,1,2,1,3,1,4,1,5,1,6,

2,2,2,3,2,4,2,5,2,6,

3,3,3,4,3,5,3,6,

4,4,4,5,4,6,

5,5,5,6,

6,6],

I,G,R,A),

term\_str(li3,I,SI),

term\_str(li3,G,SG),

term\_str(li3,R,SR),

term\_str(li3,A,SA),

format(S1,"G: %,\n A: %,\n M: %,\n E: %",SI, SG, SR, SA),

dlg\_MessageBox(

"Result",

S1,

mesbox\_iconInformation,

mesbox\_buttonsOK,

mesbox\_defaultFirst,

mesbox\_suspendApplication

),

!.

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Invoking on-line Help

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

project\_ShowHelpContext(HelpTopic):-

vpi\_ShowHelpContext("lab7.hlp",HelpTopic).

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Main Goal

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

goal

ifdef use\_mdi

vpi\_SetAttrVal(attr\_win\_mdi,b\_true),

enddef

ifdef ws\_win

ifdef use\_3dctrl

vpi\_SetAttrVal(attr\_win\_3dcontrols,b\_true),

enddef

enddef

vpi\_Init(task\_win\_Flags,task\_win\_eh,task\_win\_Menu,"lab7",task\_win\_Title).

Обоснование:

В разделе goal вызывается предикат vip\_Init(…), который запускает систему управления окнами и создает главное окно приложения.

Вызов каждой из программ, реализованных в лабораторной работе №6, осуществляется с помощью взаимодействия пользователя с пользовательским интерфейсом, созданным средствами Visual Prolog 5.2. Файлы с исходным кодом программ включались в проект при помощи директивы препроцессора **include**.

В данном задании был использован «,!» механизм отсечения в конце и в начале вызовов обработчиков событий. Использование этого отсечения в начале правила гарантирует запоминание всех целей как неизменяемых, которые были установлены перед вхождением в это правило. Использование отсечения в конце «,!.» подтверждает правильность выбора правила и прекращает дальнейший поиск.

# Задание 2

Написать для игры «Крестики-нолики» процедуру «Следующий ход», которая для заданного положения на доске находила бы наилучший ход, обеспечивающий либо предотвращение проигрыша, либо выигрыш, либо наилучший прогнозируемый результат.

Программный код:

domains

slst=symbol\*

predicates

%вставляет 0 в первый свободный элемент списка

nondeterm near(slst,slst)

%символы образуют линию

nondeterm line(string,string,string,string,string,string,string,string,string,string)

%если есть линия из двух крестиков или ноликов и третье место свободно, поставить 0

nondeterm put(string,string,string,string,string,string)

%ход

nondeterm makego(string,string,string,string,string,string,string,string,string,string,string,string,string,string,string,string,string,string,integer)

%проверка выигрыша

nondeterm checkwin(string,string,string,string,string,string,string,string,string,integer)

clauses

%символы образуют линию

line(C,S1,S2,S3,S4,S5,S6,S7,S8,S9):-

S1=C,S2=C,S3=C;

S4=C,S5=C,S6=C;

S7=C,S8=C,S9=C;

S1=C,S4=C,S7=C;

S2=C,S5=C,S8=C;

S3=C,S6=C,S9=C;

S1=C,S5=C,S9=C;

S3=C,S5=C,S7=C.

%если есть линия из двух крестиков или ноликов и третье месть свободно, то предикат вставляет 0 в пустое поле

put(S1,S2,S3,Q1,Q2,Q3):-

S1="X", S2="X", S3="", Q1=S1, Q2=S2, Q3="0",!;

S1="X", S2="", S3="X", Q1=S1, Q2="0", Q3=S3, !;

S1="" , S2="X", S3="X", Q1="0", Q2=S2, Q3=S3,!;

S1="0", S2="0", S3="", Q1=S1, Q2=S2, Q3="0",!;

S1="0", S2="", S3="0", Q1=S1, Q2="0", Q3=S3, !;

S1="" , S2="0", S3="0", Q1="0", Q2=S2, Q3=S3,!;

!, fail.

%вставляет 0 в первый свободный элемент списка

near([],[]):- !, fail.

near([H1|T1],Res):-

H1="", Res=["0"|T1],

!;

near(T1,RT),

Res=[H1|RT].

checkwin(S1,S2,S3,S4,S5,S6,S7,S8,S9,Res):-

line("0",S1,S2,S3,S4,S5,S6,S7,S8,S9),

Res=2,!;

Res=0.

%ход

makego(S1,S2,S3,S4,S5,S6,S7,S8,S9,Q1,Q2,Q3,Q4,Q5,Q6,Q7,Q8,Q9,Res):-

line("X",S1,S2,S3,S4,S5,S6,S7,S8,S9),Res=1, Q1=S1, Q2=S2, Q3=S3, Q4=S4, Q5=S5, Q6=S6, Q7=S7, Q8=S8, Q9=S9, !;

S1<>"",S2<>"",S3<>"",S4<>"",S5<>"",S6<>"",S7<>"",S8<>"",S9<>"",

Res=3, Q1=S1, Q2=S2, Q3=S3, Q4=S4, Q5=S5, Q6=S6, Q7=S7, Q8=S8, Q9=S9,!;

%проверка на то что два крестика или нолика стоят в ряд

put(S1,S2,S3,Q1,Q2,Q3), Q4=S4, Q5=S5, Q6=S6, Q7=S7, Q8=S8, Q9=S9, checkwin(Q1,Q2,Q3,Q4,Q5,Q6,Q7,Q8,Q9,Res), !;

put(S4,S5,S6,Q4,Q5,Q6), Q1=S1, Q2=S2, Q3=S3, Q7=S7, Q8=S8, Q9=S9, checkwin(Q1,Q2,Q3,Q4,Q5,Q6,Q7,Q8,Q9,Res), !;

put(S7,S8,S9,Q7,Q8,Q9), Q1=S1, Q2=S2, Q3=S3, Q4=S4, Q5=S5, Q6=S6, checkwin(Q1,Q2,Q3,Q4,Q5,Q6,Q7,Q8,Q9,Res), !;

put(S1,S4,S7,Q1,Q4,Q7), Q2=S2, Q3=S3, Q5=S5, Q6=S6, Q8=S8, Q9=S9, checkwin(Q1,Q2,Q3,Q4,Q5,Q6,Q7,Q8,Q9,Res), !;

put(S2,S5,S8,Q2,Q5,Q8), Q1=S1, Q3=S3, Q4=S4, Q6=S6, Q7=S7, Q9=S9, checkwin(Q1,Q2,Q3,Q4,Q5,Q6,Q7,Q8,Q9,Res), !;

put(S3,S6,S9,Q3,Q6,Q9), Q1=S1, Q2=S2, Q4=S4, Q5=S5, Q7=S7, Q8=S8, checkwin(Q1,Q2,Q3,Q4,Q5,Q6,Q7,Q8,Q9,Res), !;

put(S1,S5,S9,Q1,Q5,Q9), Q2=S2, Q3=S3, Q4=S4, Q6=S6, Q7=S7, Q8=S8, checkwin(Q1,Q2,Q3,Q4,Q5,Q6,Q7,Q8,Q9,Res), !;

put(S3,S5,S7,Q3,Q5,Q7), Q1=S1, Q2=S2, Q4=S4, Q6=S6, Q8=S8, Q9=S9, checkwin(Q1,Q2,Q3,Q4,Q5,Q6,Q7,Q8,Q9,Res), !;

%вставка нолика в свободное поле возле другого нолика

S1="0", near([S2,S3,S4,S5,S7,S9],[Q2,Q3,Q4,Q5,Q7,Q9|\_]), Q1=S1, Q6=S6, Q8=S8, Res=0, !;

S2="0", near([S1,S3,S5,S8],[Q1,Q3,Q5,Q8|\_]), Q2=S2, Q4=S4, Q6=S6, Q7=S7, Q9=S9, Res=0, !;

S3="0", near([S1,S2,S5,S6,S7,S9],[Q1,Q2,Q5,Q6,Q7,Q9|\_]), Q3=S3, Q4=S4, Q8=S8, Res=0, !;

S4="0", near([S1,S7,S5,S6],[Q1,Q7,Q5,Q6|\_]), Q4=S4, Q2=S2, Q3=S3, Q8=S8, Q9=S9, Res=0, !;

S5="0", near([S1,S2,S3,S4,S6,S7,S8,S9],[Q1,Q2,Q3,Q4,Q6,Q7,Q8,Q9|\_]), Q5=S5, Res=0, !;

S6="0", near([S3,S4,S5,S9],[Q3,Q4,Q5,Q9|\_]), Q6=S6, Q1=S1, Q2=S2, Q7=S7, Q8=S8, Res=0, !;

S7="0", near([S1,S3,S4,S5,S8,S9],[Q1,Q3,Q4,Q5,Q8,Q9|\_]), Q7=S7, Q2=S2, Q6=S6, Res=0, !;

S8="0", near([S2,S5,S7,S9],[Q2,Q5,Q7,Q9|\_]), Q8=S8, Q1=S1, Q3=S3, Q4=S4, Q6=S6, Res=0, !;

S9="0", near([S1,S3,S5,S6,S7,S8],[Q1,Q3,Q5,Q6,Q7,Q8|\_]), Q9=S9, Q2=S2, Q4=S4, Res=0, !;

%вставка нолика в свободное поле

S1="", Q1="0", Q2=S2, Q3=S3, Q4=S4, Q5=S5, Q6=S6, Q7=S7, Q8=S8, Q9=S9, Res=0, !;

S2="", Q1=S1, Q2="0",Q3=S3, Q4=S4, Q5=S5, Q6=S6, Q7=S7, Q8=S8, Q9=S9, Res=0, !;

S3="", Q1=S1, Q2=S2, Q3="0",Q4=S4, Q5=S5, Q6=S6, Q7=S7, Q8=S8, Q9=S9, Res=0, !;

S4="", Q1=S1, Q2=S2, Q3=S3, Q4="0",Q5=S5, Q6=S6, Q7=S7, Q8=S8, Q9=S9, Res=0, !;

S5="", Q1=S1, Q2=S2, Q3=S3, Q4=S4, Q5="0",Q6=S6, Q7=S7, Q8=S8, Q9=S9, Res=0, !;

S6="", Q1=S1, Q2=S2, Q3=S3, Q4=S4, Q5=S5, Q6="0",Q7=S7, Q8=S8, Q9=S9, Res=0, !;

S7="", Q1=S1, Q2=S2, Q3=S3, Q4=S4, Q5=S5, Q6=S6, Q7="0",Q8=S8, Q9=S9, Res=0, !;

S8="", Q1=S1, Q2=S2, Q3=S3, Q4=S4, Q5=S5, Q6=S6, Q7=S7, Q8="0",Q9=S9, Res=0, !;

S9="", Q1=S1, Q2=S2, Q3=S3, Q4=S4, Q5=S5, Q6=S6, Q7=S7, Q8=S8, Q9="0",Res=0.

Обоснование:

Работа алгоритма реализована на основе проверки всех 9-ти возможных вариантов.

Механизм отсечения используется для приостановки поиска. Работа отсечений здесь равносильна следующему заявлению: “если идет поиск вынужденных ходов, то важно найти только первое решение.”

# Задание 3

Написать программу, реализующую выигрышную стратегию для игры «Крестики-нолики» на доске 3 × 3. Игровое поле и весь процесс игры должен отображаться на экране в графическом режиме.

Программный код:

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Creation and event handling for dialog: xo\_game

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

constants

%BEGIN xo\_game, CreateParms

dlg\_xo\_game\_ResID = idd\_crosses

dlg\_xo\_game\_DlgType = wd\_Modal

dlg\_xo\_game\_Help = idh\_contents

%END xo\_game, CreateParms

predicates

dlg\_xo\_game\_eh : EHANDLER

nondeterm message(integer,WINDOW)

nondeterm go(WINDOW)

clauses

dlg\_xo\_game\_Create(Parent):-

win\_CreateResDialog(Parent,dlg\_xo\_game\_DlgType,dlg\_xo\_game\_ResID,dlg\_xo\_game\_eh,0).

%click events

dlg\_xo\_game\_eh(\_Win,e\_Control(idc\_1,\_CtrlType,\_CtrlWin,\_CtlInfo),0):-

!,

Han=win\_GetCtlHandle(\_Win,idc\_1),

win\_GetText(Han)="",

win\_SetText(Han,"X"),

go(\_Win),

!.% if field is clear then set X to there

dlg\_xo\_game\_eh(\_Win,e\_Control(idc\_2,\_CtrlType,\_CtrlWin,\_CtlInfo),0):-

!,

Han=win\_GetCtlHandle(\_Win,idc\_2),

win\_GetText(Han)="",

win\_SetText(Han,"X"),

go(\_Win),

!.

dlg\_xo\_game\_eh(\_Win,e\_Control(idc\_3,\_CtrlType,\_CtrlWin,\_CtlInfo),0):-!,

Han=win\_GetCtlHandle(\_Win,idc\_3),

win\_GetText(Han)="",

win\_SetText(Han,"X"),

go(\_Win),

!.

dlg\_xo\_game\_eh(\_Win,e\_Control(idc\_4,\_CtrlType,\_CtrlWin,\_CtlInfo),0):-!,

Han=win\_GetCtlHandle(\_Win,idc\_4),

win\_GetText(Han)="",

win\_SetText(Han,"X"),

go(\_Win),

!.

dlg\_xo\_game\_eh(\_Win,e\_Control(idc\_5,\_CtrlType,\_CtrlWin,\_CtlInfo),0):-!,

Han=win\_GetCtlHandle(\_Win,idc\_5),

win\_GetText(Han)="",

win\_SetText(Han,"X"),

go(\_Win),

!.

dlg\_xo\_game\_eh(\_Win,e\_Control(idc\_6,\_CtrlType,\_CtrlWin,\_CtlInfo),0):-!,

Han=win\_GetCtlHandle(\_Win,idc\_6),

win\_GetText(Han)="",

win\_SetText(Han,"X"),

go(\_Win),

!.

dlg\_xo\_game\_eh(\_Win,e\_Control(idc\_7,\_CtrlType,\_CtrlWin,\_CtlInfo),0):-!,

Han=win\_GetCtlHandle(\_Win,idc\_7),

win\_GetText(Han)="",

win\_SetText(Han,"X"),

go(\_Win),

!.

dlg\_xo\_game\_eh(\_Win,e\_Control(idc\_8,\_CtrlType,\_CtrlWin,\_CtlInfo),0):-!,

Han=win\_GetCtlHandle(\_Win,idc\_8),

win\_GetText(Han)="",

win\_SetText(Han,"X"),

go(\_Win),

!.

dlg\_xo\_game\_eh(\_Win,e\_Control(idc\_9,\_CtrlType,\_CtrlWin,\_CtlInfo),0):-!,

Han=win\_GetCtlHandle(\_Win,idc\_9),

win\_GetText(Han)="",

win\_SetText(Han,"X"),

go(\_Win),

!.

dlg\_xo\_game\_eh(\_,\_,\_):-

!,fail.

% computer bot logic

go(\_Win):-

% get events information

Han1=win\_GetCtlHandle(\_Win,idc\_1), S1=win\_GetText(Han1),

Han2=win\_GetCtlHandle(\_Win,idc\_2), S2=win\_GetText(Han2),

Han3=win\_GetCtlHandle(\_Win,idc\_3), S3=win\_GetText(Han3),

Han4=win\_GetCtlHandle(\_Win,idc\_4), S4=win\_GetText(Han4),

Han5=win\_GetCtlHandle(\_Win,idc\_5), S5=win\_GetText(Han5),

Han6=win\_GetCtlHandle(\_Win,idc\_6), S6=win\_GetText(Han6),

Han7=win\_GetCtlHandle(\_Win,idc\_7), S7=win\_GetText(Han7),

Han8=win\_GetCtlHandle(\_Win,idc\_8), S8=win\_GetText(Han8),

Han9=win\_GetCtlHandle(\_Win,idc\_9), S9=win\_GetText(Han9),

% algorithm

makego(S1,S2,S3,S4,S5,S6,S7,S8,S9,Q1,Q2,Q3,Q4,Q5,Q6,Q7,Q8,Q9,Res),

% setup new text

win\_SetText(Han1,Q1),

win\_SetText(Han2,Q2),

win\_SetText(Han3,Q3),

win\_SetText(Han4,Q4),

win\_SetText(Han5,Q5),

win\_SetText(Han6,Q6),

win\_SetText(Han7,Q7),

win\_SetText(Han8,Q8),

win\_SetText(Han9,Q9),

message(Res,\_Win).

message(Res,\_Win):-

Res=1,

dlg\_MessageBox(

"End game!",

"You win!",

mesbox\_iconInformation,

mesbox\_buttonsOK,

mesbox\_defaultFirst, mesbox\_suspendApplication

),

win\_Destroy(\_Win),

!;

Res=2,

dlg\_MessageBox(

"End game!",

"You lose!",

mesbox\_iconInformation,

mesbox\_buttonsOK,

mesbox\_defaultFirst,

mesbox\_suspendApplication

),

win\_Destroy(\_Win),

!;

Res=3, dlg\_MessageBox(

"End game!",

"Draw. Nobody won",

mesbox\_iconInformation,

mesbox\_buttonsOK,

mesbox\_defaultFirst, mesbox\_suspendApplication

),

win\_Destroy(\_Win),

!;

!.

Обоснование:

Для реализации игры было разработано диалоговое окно, содержащее в себе 9 кнопок.

На каждую кнопку устанавливается соответствующий обработчик событий dlg\_xo\_game\_eh, который отслеживает нажатия игрока. После нажатия пользователя на кнопку в нее записывается X (крестик) с помощью функции win\_SetText(), и ходит бот go().

В конце игры пользователя информирует информационное окно, о результатах игры, созданное с помощью функции dlg\_MessageBox().

Механизм отсечения использовался для вывода диалоговых окон (для прекращения дальнейшего поиска) и для обработчиков событий как и в первом задании.

Реализованная выигрышная стратегия:

Основная идея заключается в проверке 9-ти возможных вариантов и комбинаций (все линии по вертикали, горизонтали и диагонали). Если на одной из линий образуется два одинаковых символа, то бот ставит «0» для выигрыша или же предотвращения проигрыша. Если же данная ситуация не обнаружена, то бот ставит «0» в свободное поле, рядом с другим «0». Если же данный ход является первым для бота, то он ставит «0» в свободное место. Игра заканчивается либо победой пользователя, либо победой бота. Также может быть ничья, ситуация, когда всё поле уже заполнено, но ни один из игроков не победил.

# Выводы по проведенным экспериментам

В ходе выполнения задач лабораторной работы, я изучил работу механизма отсечения. Механизм отсечения позволяет экономить память и время на вычисления альтернативных вариантов.