МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ

РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ

ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«ОРЛОВСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

ИМЕНИ И.С. ТУРГЕНЕВА»

Кафедра информационных систем

**ОТЧЕТ**

по лабораторной работе № 7

на тему: «Многопоточное программирование. Проектирование интерфейса с использованием сети Петри»

по дисциплине: «Проектирование пользовательского интерфейса»

Выполнили: Кожухова О.А., Шорин В.Д.

Институт приборостроения, автоматизации и информационных технологий

Направление: 09.03.04 «Программная инженерия»

Группа: 71-ПГ

Проверили: Олькина Е.В., Лысков О.Э.

Отметка о зачете:

Дата: «\_\_\_\_» \_\_\_\_\_\_\_\_\_\_ 2019 г.

Орел, 2019 г

**Задание на лабораторную работу:**

Реализовать Windows приложение, которое организовывает работу двух потоков в процессе, первый из которых отвечает за бегущую строку в полосе статуса Windows окна, второй – отвечает за последовательную загрузку изображений в главную область окна. Каждый поток должна останавливать одна кнопка и с помощью нее возобновлять поток. Реализовать интерфейс с помощью сети Петри как в виде графа, так и в коде.

**Выполнение работы:**

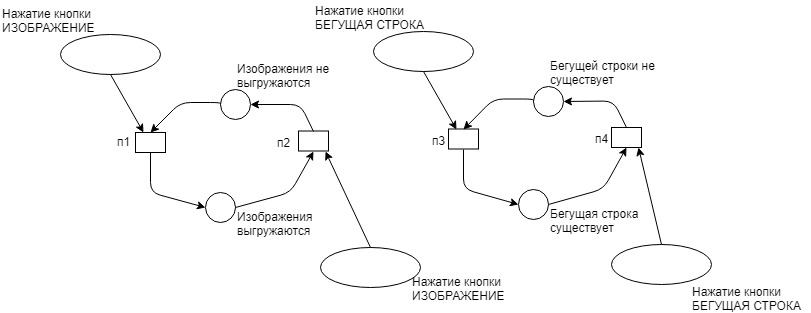


Рисунок 1 – Графическое представление сети Петри

#include "windows.h"

#include <string>

#include <iostream>

#include <vector>

#include <commctrl.h>

#include <process.h>

#define BTN\_STOP\_IMAGE 101

#define BTN\_STOP\_LINE 102

#define ID\_STATUSBAR 103

using namespace std;

BOOL RegClass(WNDPROC, LPCTSTR, UINT);

LRESULT CALLBACK WndProc(HWND, UINT, WPARAM, LPARAM);

DWORD WINAPI ThreadImage(LPVOID);

DWORD WINAPI ThreadRunnigLine(LPVOID);

HWND hwnd;

HINSTANCE hInstance;

char szMainClass[] = "MainClass";

char szTitle[] = "Lab7";

int WINAPI WinMain(HINSTANCE hInst, HINSTANCE hPrevInstance, LPSTR lpCmdLine, int nCmdShow){

MSG msg;

hInstance = hInst;

if (!RegClass(WndProc, szMainClass, COLOR\_WINDOW)) { return FALSE; }

hwnd = CreateWindow(szMainClass,

szTitle,

WS\_MINIMIZEBOX | WS\_VISIBLE | WS\_SYSMENU,

CW\_USEDEFAULT, CW\_USEDEFAULT,

500, 400,

0, 0, hInstance, NULL);

if (!hwnd) { return FALSE; }

while (GetMessage(&msg, 0, 0, 0)){

TranslateMessage(&msg);

DispatchMessage(&msg);

}

return msg.wParam;

}

BOOL RegClass(WNDPROC Proc, LPCTSTR szName, UINT brBackground){

WNDCLASS wc;

wc.style = 0;

wc.cbClsExtra = 0;

wc.cbWndExtra = 0;

wc.lpfnWndProc = Proc;

wc.hInstance = hInstance;

wc.hIcon = LoadIcon(NULL, IDI\_APPLICATION);

wc.hCursor = LoadCursor(NULL, IDC\_ARROW);

wc.hbrBackground = (HBRUSH)(brBackground + 1);

wc.lpszMenuName = (LPCTSTR)NULL;

wc.lpszClassName = szName;

return (RegisterClass(&wc) != 0);

}

HWND hwndBtnStopImage, hwndBtnStopLine;

HWND hwndStatusBar;

HWND hwndImage;

HANDLE hThreadImage;

HANDLE hThreadRunnigLine;

bool isImgPressed = false;

bool isLinePressed = false;

LRESULT CALLBACK WndProc(HWND hWnd, UINT msg, WPARAM wParam, LPARAM lParam){

switch (msg){

case WM\_CREATE:

{

hwndBtnStopImage = CreateWindow("button", "IMAGE",

WS\_CHILD | WS\_VISIBLE | BS\_DEFPUSHBUTTON,

20, 20,

100, 30,

hWnd, (HMENU) BTN\_STOP\_IMAGE, hInstance, NULL);

hwndBtnStopLine = CreateWindow("button", "LINE",

WS\_CHILD | WS\_VISIBLE | BS\_DEFPUSHBUTTON,

20, 70,

100, 30,

hWnd, (HMENU) BTN\_STOP\_LINE, hInstance, NULL);

hwndStatusBar = CreateStatusWindow(WS\_CHILD | WS\_VISIBLE,

"My Status Bar",

hWnd,

ID\_STATUSBAR);

hThreadImage = CreateThread(NULL, 0, ThreadImage, 0, CREATE\_SUSPENDED, 0);

if(hThreadImage == NULL){

MessageBox(hWnd, "CreateThread hThreadImage failed", "Err", MB\_OK);

}

hThreadRunnigLine = CreateThread(NULL, 0, ThreadRunnigLine, 0, CREATE\_SUSPENDED, 0);

if(hThreadRunnigLine == NULL){

MessageBox(hWnd, "CreateThread hThreadRunnigLine failed", "Err", MB\_OK);

}

return 0;

}

case WM\_COMMAND:

{

switch (LOWORD(wParam)){

case BTN\_STOP\_IMAGE:

{

if(isImgPressed){

SuspendThread(hThreadImage);

}

else{

ResumeThread(hThreadImage);

}

isImgPressed = !isImgPressed;

break;

}

case BTN\_STOP\_LINE:

{

if(isLinePressed){

SuspendThread(hThreadRunnigLine);

}

else{

ResumeThread(hThreadRunnigLine);

}

isLinePressed = !isLinePressed;

break;

}

}

break;

}

case WM\_DESTROY:

{

PostQuitMessage(0);

return 0;

}

}

return DefWindowProc(hWnd, msg, wParam, lParam);

}

PAINTSTRUCT ps;

HDC hdc, hCompatibleDC;

HANDLE hBitMap, hOldBitMap;

RECT rect;

BITMAP bitMap;

int curImg = 0;

vector<string> images = {"img1.bmp", "img2.bmp", "img3.bmp"};

DWORD WINAPI ThreadImage(LPVOID lpParameter){

while(true){

hdc = GetDC(hwnd);

hBitMap = LoadImage(NULL, images[curImg].c\_str(), IMAGE\_BITMAP, 0, 0, LR\_LOADFROMFILE);

GetObject(hBitMap, sizeof(BITMAP), &bitMap);

hCompatibleDC = CreateCompatibleDC(hdc);

hOldBitMap = SelectObject(hCompatibleDC, hBitMap);

GetClientRect(hwnd, &rect);

StretchBlt(hdc, 120, 10, rect.right, rect.bottom,

hCompatibleDC, 0, 0, 800, 800,

SRCCOPY);

SelectObject(hCompatibleDC, hOldBitMap);

DeleteObject(hBitMap);

DeleteDC(hCompatibleDC);

ReleaseDC(hwnd, hdc);

curImg++;

if(curImg == 3){

curImg = 0;

}

Sleep(2000);

}

return 0;

}

string strName = "My Status Bar";

string str;

int countSpaces = 0;

DWORD WINAPI ThreadRunnigLine(LPVOID lpParameter){

while(true){

str = "";

for(int i = 0; i < countSpaces; i++){

str.insert(0, " ");

}

str += strName;

countSpaces++;

if(countSpaces == 165){

str = "";

countSpaces = 0;

}

SetWindowText(hwndStatusBar, str.c\_str());

Sleep(100);

}

return 0;

}