תכנות מונחה עצמים ב# C

מטלה 4 –
GUI

מגישים:

אושרי בוסקילה	313349193	oshrib297@gmail.com
נועה בר-דוד	205788565	noaaruppin@gmail.com

שאלה 1:

Program

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace GUI_Q1
{
    internal static class Program
    {
        /// <summary>
        /// The main entry point for the application.
        /// </summary>
        [STAThread]
        static void Main()
        {
            Application.EnableVisualStyles();
            Application.SetCompatibleTextRenderingDefault(false);
            Application.Run(new Form2());
        }
    }
}
```

DataFile

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace HW4
{
    enum FileTypeExtension { TXT = 0, DOC = 1, DOCX = 2, PDF = 3, PPTX = 4
};
    internal class DataFile
    {
        private string FileName;
        private DateTime lastUpadateTime;
        private string Data;
        private FileTypeExtension type ;
        private static int counter;


        public string getFileName()
        {
            return this.FileName;
        }
        public bool setFileName(string name)
        {
            for (int i = 0; i < name.Length; i++)
            {
                if (name[i] == '?' ||
                    name[i] == '/' ||
                    name[i] == '*' ||
                    name[i] == '>' ||
                    name[i] == '<' ||
                    name[i] == ':' ||
                    name[i] == '"'
                    )
                {
                    MessageBox.Show("A file cannot contain signs other
than letters!", "Error", MessageBoxButtons.OK, MessageBoxIcon.Error);
                    return false;
                }

            }

            this.FileName = name;
            return true;
        }
        public string getData()
        {
            return this.Data;
        }
        public void setData(string name)
        {
            this.Data = name;
        }
        public void SetTime()
        {
            this.lastUpadateTime = DateTime.Now;
        }
        public DateTime GetTime()
```

```
        {
            return this.lastUpadateTime;
        }
        public FileTypeExtension getType()
        {
            return this.type;
        }
        public DataFile(string newName, string newData, FileTypeExtension
typeExtension)
        {
            this.type = typeExtension;
            setFileName(newName);
            setData(newData);
            SetTime();
        }
        public DataFile() :this("sameFile"+ counter,
"",FileTypeExtension.TXT) { counter++; }
        public DataFile(DataFile oldFile)
        {
            this.FileName = oldFile.getFileName() + "1";
            this.Data = oldFile.getData();
            this.SetTime();
        }
        public int GetSize()
        {
           ;  return this.Data.Length;
        }

        public string dir()
        {
           return string.Format("{0} {1} KB {2}. {3}\n", GetTime(),
String.Format("{0:0.00}", (GetSize() / 1024)), getFileName(), this.type);
        }
}
}
```

CompareFiles

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace HW4
{
    static class CompareFiles
    {

        public static bool EqualsFiles(DataFile f1, DataFile f2)
        {
            return (f1.getData() == f2.getData() && f1.getFileName() ==
f2.getFileName());

        }
        public static int CompareSizeFiles(DataFile f1, DataFile f2)
        {

            if (f1.GetSize() > f2.GetSize() )
            {
                return 1;
            }
            if (f2.GetSize() > f1.GetSize())
            {
                return -1;
            }
            return 0;
        }

    }


}
```

QueueFiles

```
using System;
using System.Collections.Generic;
using System.IO;
using System.Linq;
using System.Runtime.InteropServices.WindowsRuntime;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace HW4
{
    internal class QueueFiles
    {
        DataFile[] queue;
        int freeIndex;

        public QueueFiles ()
        {
            this.queue = new DataFile[0];
            freeIndex = -1;
        }
        public bool isEmpty()
        {

            if ( freeIndex == -1 && this.queue == null|| queue.Length==0)
            {

                return true;
            }
            return false;
        }

        public void enQueue (DataFile newfile)
        {

            DataFile[] newqueue = new DataFile[queue.Length+1];
           if(queue.Length == 0 )
            {
                queue = newqueue;
                newqueue[0] = newfile;
                freeIndex++;
                return;
            }
           for(int i = 0; i < queue.Length; i++)
            {

                if (CompareFiles.EqualsFiles(queue[i],newfile) )
                {

                    MessageBox.Show("The file already exist in queue");
                    return;
                }
                newqueue[i] = queue[i];
            }
            newqueue[queue.Length] = newfile;
            queue = newqueue;
            freeIndex++;
        }
        public DataFile Dequeue ()
        {
```

```
            DataFile[] newqueue = new DataFile[queue.Length-1];
            for(int i = 1; i < queue.Length; i++ )
            {
                newqueue[i - 1] = queue[i];
            }
            DataFile temp = queue[0];
            freeIndex = queue.Length - 1;
            queue = newqueue;
            return temp;


        }
        public DataFile BigFile ( )
        {
            QueueFiles temp = new QueueFiles();
            temp.enQueue(queue[0]);
            if (queue.Length == 1)
            {
                Console.WriteLine("one slot queue");
                return this.queue[0];
            }
            foreach (DataFile file in this.queue)
            {
                if (CompareFiles.CompareSizeFiles(file, temp.queue[0]) ==
1)
                    temp.queue[0] = file;
            }
            return temp.queue[0];
        }


        public string PrintQueue ( )
        {
            string print= "The files are:";
            if (queue == null)
            {
               print= "There are no details in the list";
            }
            else
            {
                foreach (DataFile file in this.queue)
                {
                    print += "\r\n" + file.dir();
                }

            }

            return print;
        }

        public QueueFiles SearchFileByType(FileTypeExtension type)
        {

            QueueFiles temp = new QueueFiles();

            if (this.isEmpty()) return null;

            foreach (DataFile file in this.queue)
            {
                if (file.getType() == type)
                {
```

```
                    temp.enQueue(file);
                }
            }
            if (!temp.isEmpty())
                return temp;
            else
                return null;
        }
    }
}
```

Form2

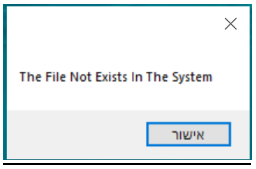
```
using HW4;
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace GUI_Q1
{
    public partial class Form2 : Form
    {

        QueueFiles list = new QueueFiles();

        public Form2()
        {
            InitializeComponent();
        }
        private void Form2_Load(object sender, EventArgs e)
        {
           MessageBox.Show("The File Not Exists In The System");
            lblload.Visible = true;

        }


        private void bbtnon1_Click(object sender, EventArgs e)
        {
            grpAddFile.Visible = false;
           // lblload.Visible = false;
            grpByType.Visible = false;
            txtPrint.Visible = false;
            DataFile file = new DataFile();
            list.enQueue(file);
            MessageBox.Show("Default file added to the queue!");

        }

        private void btnop2_Click(object sender, EventArgs e)
        {
```

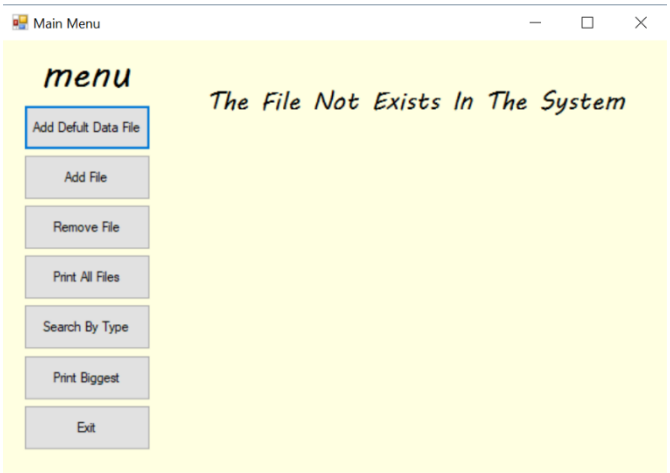
```
            // lblload.Visible = false;
            DataFile file = new DataFile();
            grpByType.Visible = false;
            txtPrint.Visible = false;
            grpAddFile.Visible = true;

        }

        private void btnCreateFile_Click(object sender, EventArgs e)
        {
            DataFile file = new DataFile();
            if (string.IsNullOrWhiteSpace(txbData.Text) ||
               string.IsNullOrWhiteSpace(txbName.Text) ||
                cmbtape.SelectedIndex == -1)
            {
                MessageBox.Show("You need to fill in all the fields!!!",
"Error", MessageBoxButtons.OK, MessageBoxIcon.Error);
            }
            else
            {
                file = new DataFile(txbName.Text, txbData.Text,
(FileTypeExtension)cmbtape.SelectedIndex);
                list.enQueue(file);
                txbName.Text = "";
                txbData.Text = "";
                cmbtape.SelectedIndex = -1;
                MessageBox.Show("File added successfully!");
                grpAddFile.Visible = false;
            }
        }

        private void btnOp3_Click(object sender, EventArgs e)
        {
            //lblload.Visible = false;
            grpAddFile.Visible = false;
            grpByType.Visible = false;
            txtPrint.Visible = false;
            if (list.isEmpty())
            {
              MessageBox.Show("The queue is empty!");
                return;
            }
            list.Dequeue();
            MessageBox.Show("Removed File!");
        }
        private void btnOp4_Click(object sender, EventArgs e)
        {
            //lblload.Visible = false;
            grpByType.Visible = false;
            txtPrint.Visible = false;

            if (list.isEmpty())
            {
                MessageBox.Show("The queue is empty!");
                return;
            }
            txtPrint.Visible = true;
            list.PrintQueue();
            txtPrint.Text = list.PrintQueue();

        }
         private void btnOp5_Click(object sender, EventArgs e)
         {
```

```
            if (list.isEmpty())
            {
                MessageBox.Show("The queue is empty!");
                return;
            }
            grpAddFile.Visible = false;
            txtPrint.Visible = false;
            grpByType.Visible = true;
        }

        private void btnSearch_Click(object sender, EventArgs e)
        {
            //lblload.Visible = false;
            grpByType.Visible=true;
            if (cmbTypeToSearch.SelectedIndex==-1)
            {
                MessageBox.Show("You must choose one of the options!");
                return;
            }

            int s = cmbTypeToSearch.SelectedIndex;
            QueueFiles q = list.SearchFileByType((FileTypeExtension)s);
            if (q == null)
            {
                txtPrint.Visible = false;
                MessageBox.Show("There are no files of the type you
selected");
                cmbTypeToSearch.SelectedIndex = -1;
                return;
            }
            txtPrint.Visible = true;
            txtPrint.Clear();
            txtPrint.Text = q.PrintQueue();
            cmbTypeToSearch.SelectedIndex = -1;

        }


        private void btnOp6_Click(object sender, EventArgs e)
        {
            //lblload.Visible = false;
            DataFile file = new DataFile();
            grpAddFile.Visible = false;
            txtPrint.Visible = false;
            grpByType.Visible = false;
            if (list.isEmpty())
            {
                MessageBox.Show("The queue is empty!");
                return;
            }
            file = list.BigFile();
            MessageBox.Show("The Biggest File Is: \n" + file.dir());
        }

        private void btnOp7_Click(object sender, EventArgs e)
        {
           // lblload.Visible = false;
            Application.Exit();
        }


    }
}
```

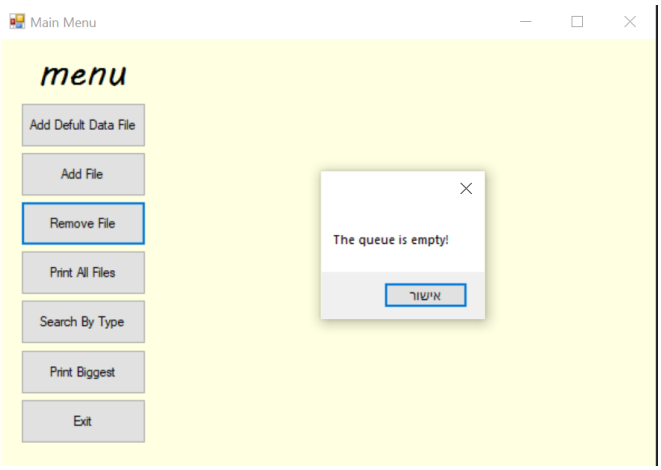
פלטים:

מסך ראשי עם פתיחת התוכנה (ברגע שהטופס נטען)

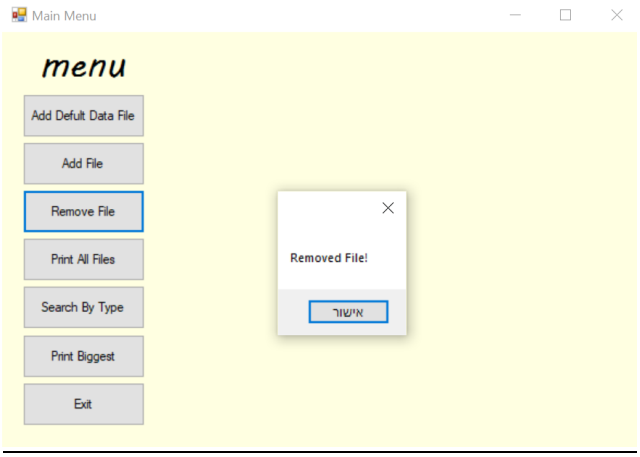




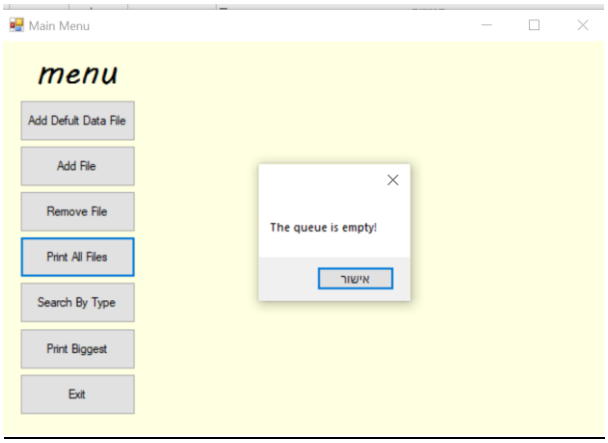
ניסיון הסרת קובץ מהתור כאשר התור ריק



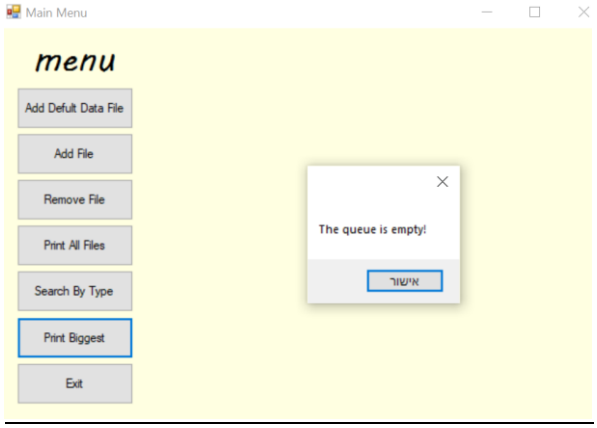
מחיקת קובץ מהרשימה



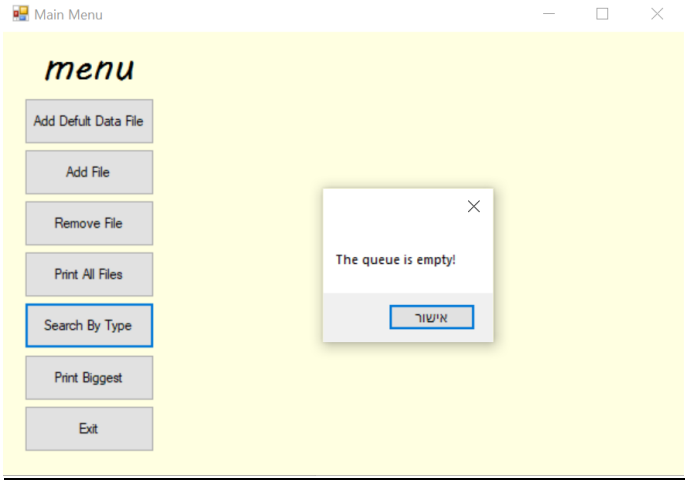
ניסיון הדפסת קבצים כאשר התור ריק



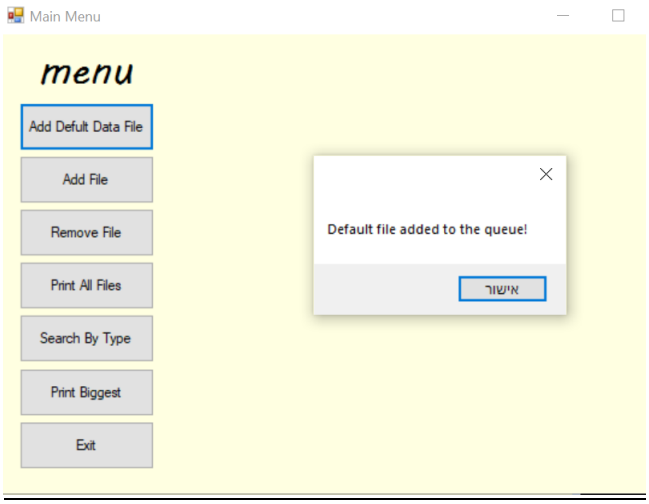
ניסיון הדפסת הקובץ הגדול ביותר הקיים בתור כאשר התור ריק



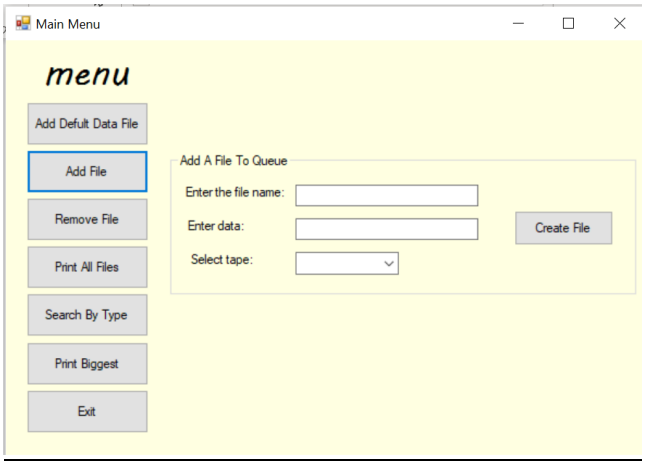
ניסיון חיפוש קובץ לפי סוג כאשר התור ריק



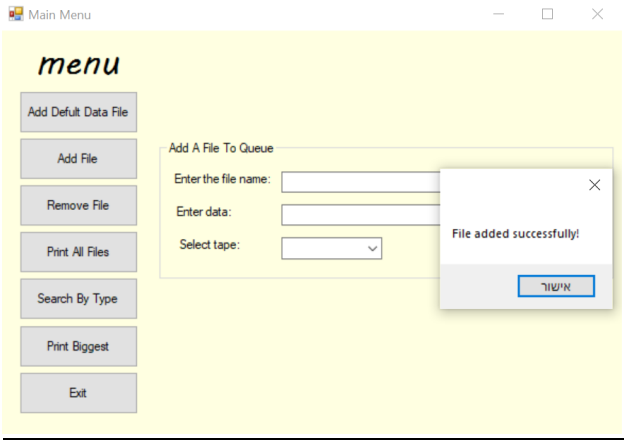
קבלת MessageBox עם הוספת קובץ לתור



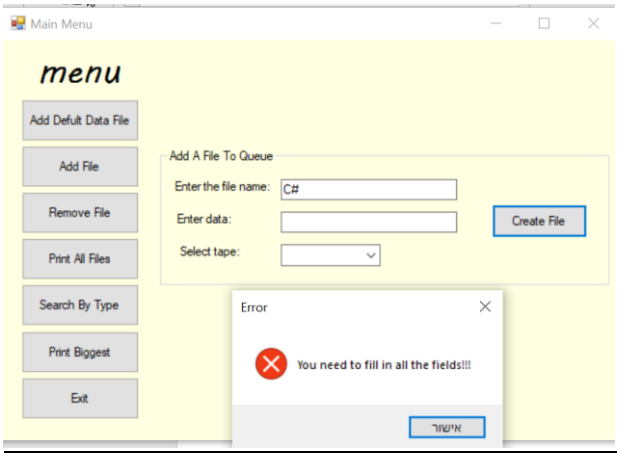
הוספת קובץ ע"י המשתמש



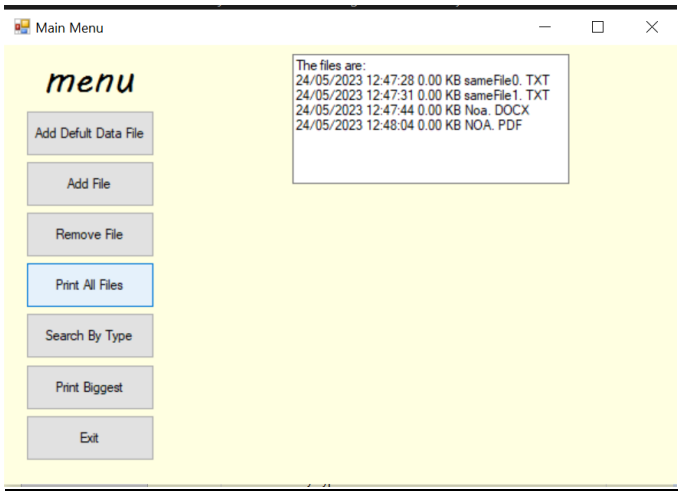
קבלת MessageBox וניקוי תאים בעת הוספת קובץ ע"י המשתמש



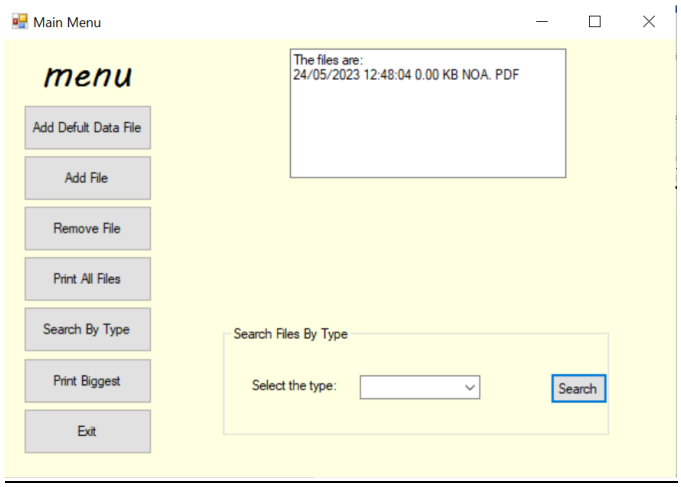
קבלת MessageBox במקרה שאחד התאים ריק



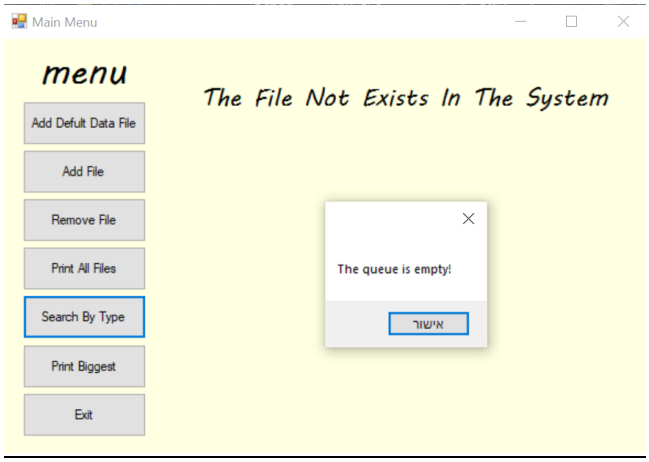
הדפסת כל הקבצים



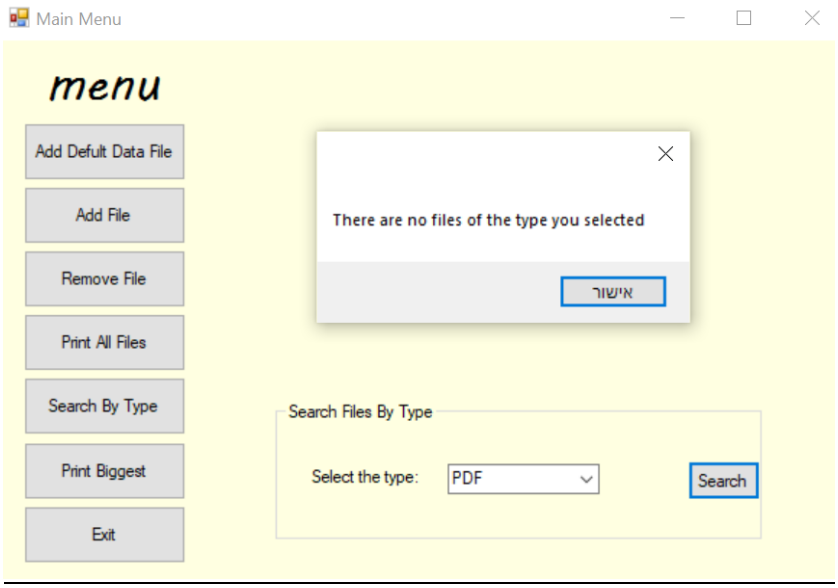
חיפוש לפי סוג



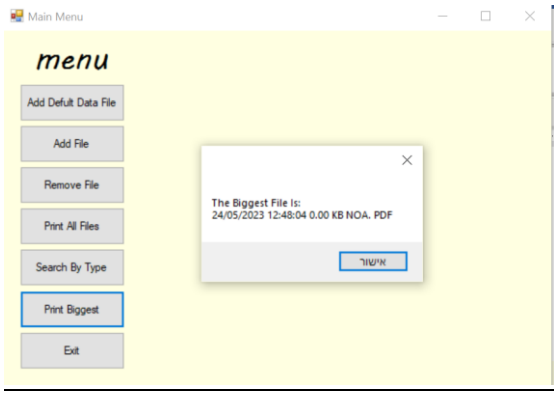
ניסיון חיפוש לפי סוג כאשר הרשימה ריקה



קבלת הודעת שגיאה כאשר מחפשים מסוג קובץ שאינו קיים ברשימה



הדפסת הקובץ הגדול ביותר



שאלה 2:

דרישות חובה למשק:

1. **שימוש בTIMER**

שינוי צבע LABLE WELCOM בכל שניה משחור לאדום.

2. **שימוש ב-5 פקדים לפחות:**

א. Button

ב. Lable

ג. Textbox

ד. Groupbox

ה. Tooltip

ו. NumericUpDown

ז. DataGridView

ח. PictureBox

3. **הצגת סכום קניה**

סכום הקניה מופיע בתוך TEXTBOX בצמוד ל-LABLE בשם "TOTAL", אותו ניתן לראות לאחר לחיצה על תמונה של סל קניות ומעבר לחלון "CART"

4. **מימוש יותר מאירוע אחד:**

א. פקד התמונה של סל הקניות picCart:

1). Click (לחיצה) - נפתח חלון סיכום ההזמנה

2). MouseHover המציג מלל

ב. הפקדים "Add to cart":

1). Click – מוסיף את המוצר לסל הקניות

2). MouseClick – משנה את צבע הפקד בעת הלחיצה

הסבר הממשק:

בשאלה זו החלטנו לתכנן וליצור חנות פרחים "אונליין" שתהיה נוחה ואינטואיטיבית ככל האפשר, בעת פתיחת התוכנה ניתן לראות שלט מהבהב מחליף צבעים (בעזרת טיימר) המברך את המשתמש ברכת ברוך הבא לחנות הפרחים של נועה ואושרי, באופן כללי באפשרות המשתמש לראות את שם הפרח, תמונה להמחשה, ובנוסף בעזרת ריחוף של העכבר מעל Groupbox הרצוי ניתן לראות את מחיר הזר.
לאחר מכן באפשרות המשתמש לתכנן את כמות הפרחים שברצונו להוסיף לעגלה, במסך זה התוכנה תציג ותחשב את סך הפריטים וסך התשלום לפי פריט באמצעות Data grid ובחלק התחתון תחשב את סך התשלום עבור כל ההזמנה. לאחר שהמשתמש מאשר את ההזמנה תופיע הודעה שהרכישה בוצעה ותוך 7 ימי עסקים המשלוח יגיע ללקוח.

פקדים ואירועים רלוונטיים:
-בעת "עמידה" על Groupbox של כל פרח יופיע מחירו ליחידה.
-הלחצן של הוספה לסל(Add to cart) מתחיל במצב לא זמין ורק לאחר הוספה של מוצר אחד לפחות מאותו סוג (באמצעות פקד NumericUpDown כל עוד גדול מ-0) לחצן ההוספה זמין.

-כל הפרחים מאותחלים עם שמותיהם ומחירם ליחידה.

-הכמות המחושבת של כל מוצר משתנה בהתאם לפקד ה- NumericUpDown

-לאחר לחיצה על פקד Add to cart נשלחים לפונקציה TotalPrice הארגומנטים של הכמות ומחיר והיא מחזירה את המחיר המחושב הסופי.

- בעת לחיצה על הוספה לסל, מתקבל MessageBox המציג את שם המוצר והכמות אותה דרש המשתמש (אישור עבורו כי המוצרים נכנסו לסל) והמידע עובר לפונקציה UpdateOrder אשר יוצר טבלת נתונים חדשה ומזינה את המידע בתאי הטבלה בהתאם לעיצוב שקבענו.

-בנוסף בתוך חלון סל הרכישה בחלקו התחתון יופיע TextBox המראה את סך התשלום בעבור כל הפריטים שהמשתמש הוסיף לעגלה.
-בכל רגע נתון יכול המשתמש לחזור למסך בו מופיע מבחר הפרחים בחנות להוסיף לעגלה פריטים והרשימה והמחיר הסופי בעמוד עגלת הרכישה יתעדכנו!

-בעת לחיצה על כפתור Finished shopping, מתקבל MessageBox למשתמש על קליטת הזמנתו והתוכנית נסגרת.

קוד

Program

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace GUI_Q2
{
    internal static class Program
    {

        /// <summary>
        /// The main entry point for the application.
        /// </summary>
        [STAThread]
        static void Main()
        {
            Application.EnableVisualStyles();
            Application.SetCompatibleTextRenderingDefault(false);
            Application.Run(new Form1());

        }
    }
}
```

Flowers

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace GUI_Q2
{
    internal class Flowers
    {
        public int TotalPrice(int price, int count)
        {
            int totalprice = (int)price * count;
            return totalprice;

        }

    }

}
```

Form1

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using static System.Windows.Forms.VisualStyles.VisualStyleElement;

namespace GUI_Q2
{

    public partial class Form1 : Form
    {

        Cart cart = new Cart();

        public Form1()
        {
            InitializeComponent();
        }
        private void picCart_Click(object sender, EventArgs e)
        {

            cart.ShowDialog();
        }

        public void btnAddSachlav_Click(object sender, EventArgs e)
        {
            Flowers flowers = new Flowers();

            string item = "Sachlav";
            int price = 10;
            int count = (int)nudSachlav.Value;
            int totalprice_Sachlav = flowers.TotalPrice(price, count);

            MessageBox.Show(count+ " \"" +item +"\"" + " Successfully
added to cart");
            nudSachlav.Enabled = false;
            cart.UpdateOrder(item, count, totalprice_Sachlav);
            btnAddSachlav.Enabled = false;
        }

        private void nudSachlav_ValueChanged(object sender, EventArgs e)
        {
            if((int)nudSachlav.Value != 0)
            {
                btnAddSachlav.Enabled = true;
            }
        }

        private void gbSachlav_MouseMove(object sender, EventArgs e)
        {
            toolTip1.SetToolTip(gbSachlav, "The price is 10$ per unit");
        }

        private void nudHarzit_ValueChanged(object sender, EventArgs e)
        {
```

```
            if ((int)nudHarzit.Value != 0)
            {
                btnAddHarzit.Enabled = true;
            }
        }

        private void btnAddHarzit_Click(object sender, EventArgs e)
        {
            Flowers flowers = new Flowers();

            string item = "Harzit";
            int price = 20;
            int count = (int)nudHarzit.Value;
            int totalprice_Harzit = flowers.TotalPrice(price, count);

            MessageBox.Show(count + " \"" + item + "\"" + " Successfully
added to cart");
            nudHarzit.Enabled = false;
            cart.UpdateOrder(item, count, totalprice_Harzit);
            btnAddHarzit.Enabled = false;

        }
        private void nudVered_ValueChanged(object sender, EventArgs e)
        {
            if ((int)nudVered.Value != 0)
            {
                btnAddVered.Enabled = true;
            }
        }

        private void btnAddVered_Click(object sender, EventArgs e)
        {
             Flowers flowers = new Flowers();

            string item = "Vered";
            int price = 30;
            int count = (int)nudVered.Value;
            int totalprice_Vered = flowers.TotalPrice(price, count);

            MessageBox.Show(count+ " \"" +item +"\"" + " Successfully
added to cart");
            nudVered.Enabled = false;
            cart.UpdateOrder(item, count, totalprice_Vered);
            btnAddVered.Enabled = false;
        }

        private void grpVered_Enter(object sender, EventArgs e)
        {

        }

        private void nudYakinton_ValueChanged(object sender, EventArgs e)
        {
            if ((int)nudYakinton.Value != 0)
            {
                btnAddYakinton.Enabled = true;
            }
        }

        private void btnAddYakinton_Click(object sender, EventArgs e)
        {
            Flowers flowers = new Flowers();
```

```
            string item = "Yakinton";
            int price = 40;
            int count = (int)nudYakinton.Value;
            int totalprice_Yakinton = flowers.TotalPrice(price, count);

            MessageBox.Show(count + " \"" + item + "\"" + " Successfully
added to cart");
            nudYakinton.Enabled = false;
            cart.UpdateOrder(item, count, totalprice_Yakinton);
            btnAddYakinton.Enabled = false;
        }

        private void nudCalanit_ValueChanged(object sender, EventArgs e)
        {
            if ((int)nudCalanit.Value != 0)
            {
                btnAddCalanit.Enabled = true;
            }
        }

        private void btnAddCalanit_Click(object sender, EventArgs e)
        {
            Flowers flowers = new Flowers();

            string item = "Calanit";
            int price = 50;
            int count = (int)nudCalanit.Value;
            int totalprice_Calanit = flowers.TotalPrice(price, count);

            MessageBox.Show(count + " \"" + item + "\"" + " Successfully
added to cart");
            nudCalanit.Enabled = false;
            cart.UpdateOrder(item, count, totalprice_Calanit);
            btnAddCalanit.Enabled = false;
        }

        private void nudNarkis_ValueChanged(object sender, EventArgs e)
        {
            if ((int)nudNarkis.Value != 0)
            {
                btnAddNarkis.Enabled = true;
            }
        }

        private void btnAddNarkis_Click(object sender, EventArgs e)
        {
            Flowers flowers = new Flowers();

            string item = "Narkis";
            int price = 60;
            int count = (int)nudNarkis.Value;
            int totalprice_Narkis = flowers.TotalPrice(price, count);

            MessageBox.Show(count + " \"" + item + "\"" + " Successfully
added to cart");
            nudNarkis.Enabled = false;
            cart.UpdateOrder(item, count, totalprice_Narkis);
            btnAddNarkis.Enabled = false;
        }

        private void nudShoshan_ValueChanged(object sender, EventArgs e)
        {
            if ((int)nudShoshan.Value != 0)
```

```
            {
                btnAddShoshan.Enabled = true;
            }
        }

        private void btnAddShoshan_Click(object sender, EventArgs e)
        {
            Flowers flowers = new Flowers();

            string item = "Shoshan";
            int price = 70;
            int count = (int)nudShoshan.Value;
            int totalprice_Shoshan = flowers.TotalPrice(price, count);

            MessageBox.Show(count + " \"" + item + "\"" + " Successfully
added to cart");
            nudShoshan.Enabled = false;
            cart.UpdateOrder(item, count, totalprice_Shoshan);
            btnAddShoshan.Enabled = false;
        }

        private void nudHamania_ValueChanged(object sender, EventArgs e)
        {
            if ((int)nudHamania.Value != 0)
            {
                btnAddHamanya.Enabled = true;
            }

        }

        private void btnAddHamanya_Click(object sender, EventArgs e)
        {
            Flowers flowers = new Flowers();

            string item = "Hamanya";
            int price = 80;
            int count = (int)nudHamania.Value;
            int totalprice_Hamanya = flowers.TotalPrice(price, count);

            MessageBox.Show(count + " \"" + item + "\"" + " Successfully
added to cart");
            nudHamania.Enabled = false;
            cart.UpdateOrder(item, count, totalprice_Hamanya);
            btnAddHamanya.Enabled=false;
        }

        private void timerColor_Tick(object sender, EventArgs e)
        {
            if (lblwelcome.ForeColor == Color.Black)
            {
                lblwelcome.ForeColor = Color.Red;
            }
            else
            {
                lblwelcome.ForeColor = Color.Black;
            }

        }

        private void btnAddVered_MouseHover(object sender, EventArgs e)
        {
            btnAddVered.BackColor = Color.LightPink;
        }
```

```
        private void grpHarzit_MouseHover(object sender, EventArgs e)
        {
            ttHarzit.SetToolTip(grpHarzit, "The price is 20$ per unit");

        }

        private void grpVered_MouseHover(object sender, EventArgs e)
        {
            ttVered.SetToolTip(grpVered, "The price is 30$ per unit");
        }

        private void grpYakinton_MouseHover(object sender, EventArgs e)
        {
            ttcalanit.SetToolTip(grpYakinton, "The price is 40$ per
unit");
        }

        private void grpCalanit_MouseHover(object sender, EventArgs e)
        {
            ttcalanit.SetToolTip(grpCalanit, "The price is 50$ per unit");
        }

        private void grpNarkis_MouseHover(object sender, EventArgs e)
        {
            ttNarkis.SetToolTip(grpNarkis, "The price is 60$ per unit");
        }

        private void grpShoshan_MouseHover(object sender, EventArgs e)
        {
            ttShoshan.SetToolTip(grpShoshan, "The price is 70$ per unit");
        }

        private void grpHamania_MouseHover(object sender, EventArgs e)
        {
            ttHamanya.SetToolTip(grpHamania, "The price is 80$ per unit");

        }

        private void picCart_MouseHover(object sender, EventArgs e)
        {
            ttcart.SetToolTip(picCart, "Click here to watch your order");
        }

        private void btnAddSachlav_MouseClick(object sender,
MouseEventArgs e)
        {
            btnAddSachlav.BackColor = Color.LightPink;
        }

        private void btnAddHarzit_MouseClick(object sender, MouseEventArgs
e)
        {
            btnAddHarzit.BackColor = Color.LightPink;
        }

        private void btnAddYakinton_MouseClick(object sender,
MouseEventArgs e)
        {
            btnAddYakinton.BackColor = Color.LightPink;

        }
```

```
        private void btnAddCalanit_MouseClick(object sender,
MouseEventArgs e)
        {
            btnAddCalanit.BackColor = Color.LightPink;
        }

        private void btnAddNarkis_MouseClick(object sender, MouseEventArgs
e)
        {
            btnAddNarkis.BackColor = Color.LightPink;
        }

        private void btnAddShoshan_MouseClick(object sender,
MouseEventArgs e)
        {
            btnAddShoshan.BackColor = Color.LightPink;
        }

        private void btnAddHamanya_MouseClick(object sender,
MouseEventArgs e)
        {
            btnAddHamanya.BackColor = Color.LightPink;
        }
    }

}
```

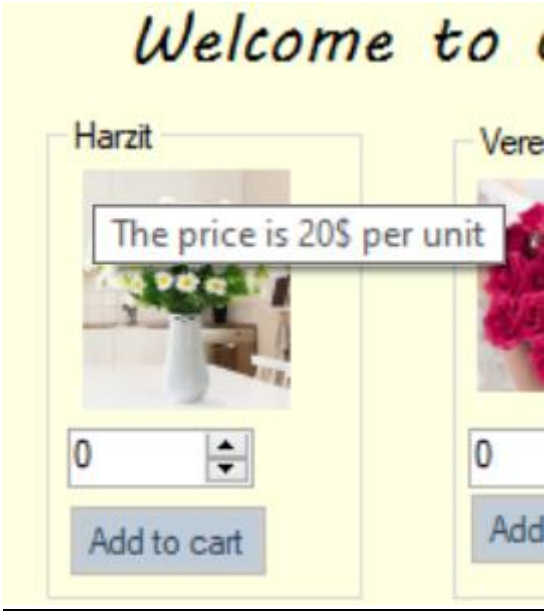
Cart

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using static
System.Windows.Forms.VisualStyles.VisualStyleElement.ProgressBar;

namespace GUI_Q2
{
    public partial class Cart : Form
    {

        public Cart()
        {
            InitializeComponent();

        }

        public void UpdateOrder(string item, int count, int totalprice)
        {
            var index = this.dataGridView1.Rows.Add();
            this.dataGridView1.Rows[index].Cells[0].Value = item;
            this.dataGridView1.Rows[index].Cells[1].Value = count;
            this.dataGridView1.Rows[index].Cells[2].Value = totalprice;
            int totalorder = 0;

            foreach (DataGridViewRow row in dataGridView1.Rows)
            {
                totalorder += (int)row.Cells[2].Value  ;
            }

            textBox1.Text = totalorder.ToString() + "$";
        }

        private void dataGridView1_CellContentClick(object sender,
DataGridViewCellEventArgs e)
        {

        }

        private void btnFinish_Click(object sender, EventArgs e)
        {
            MessageBox.Show("Your order has been received and will arrive
in 7 business days");
            Application.Exit();
        }
    }
}
```

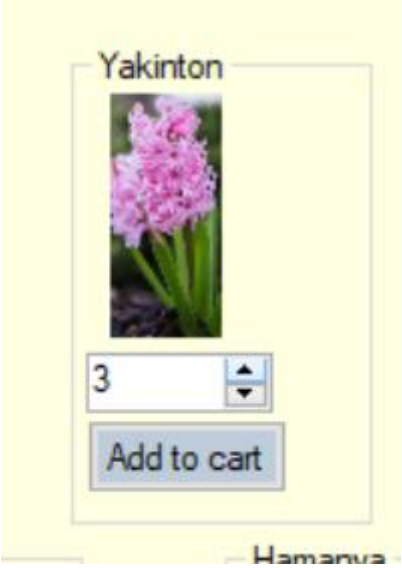
פלטים

מסך ראשי - פקדי הוספה לסל אינם זמינים

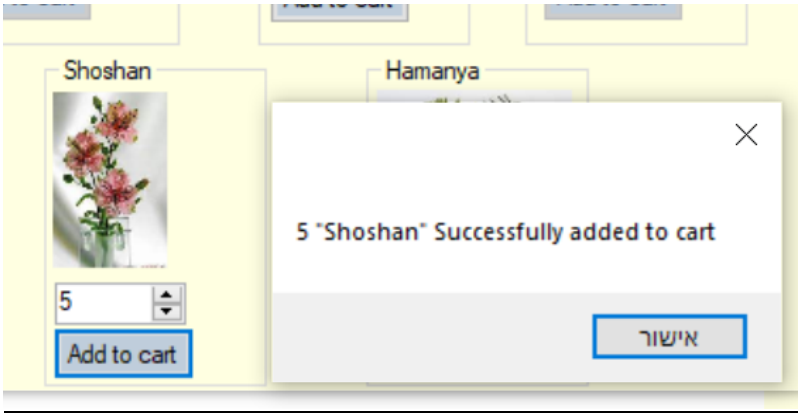
בעת עמידה על כל Groupbox של כל פרח, מופיע המחיר ליחידה:



עם הוספת מוצר באמצעות פקד NumericUpDown האפשרות להוספה לסל נפתחת:

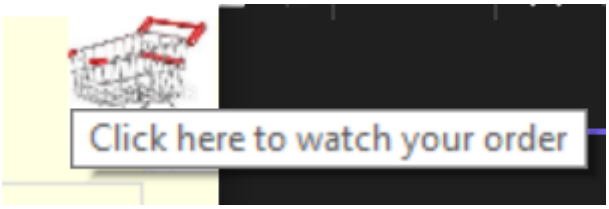


עם לחיצה על פקד ההוספה לסל מתקבלת הודעה אודות המוצר וכמותו:

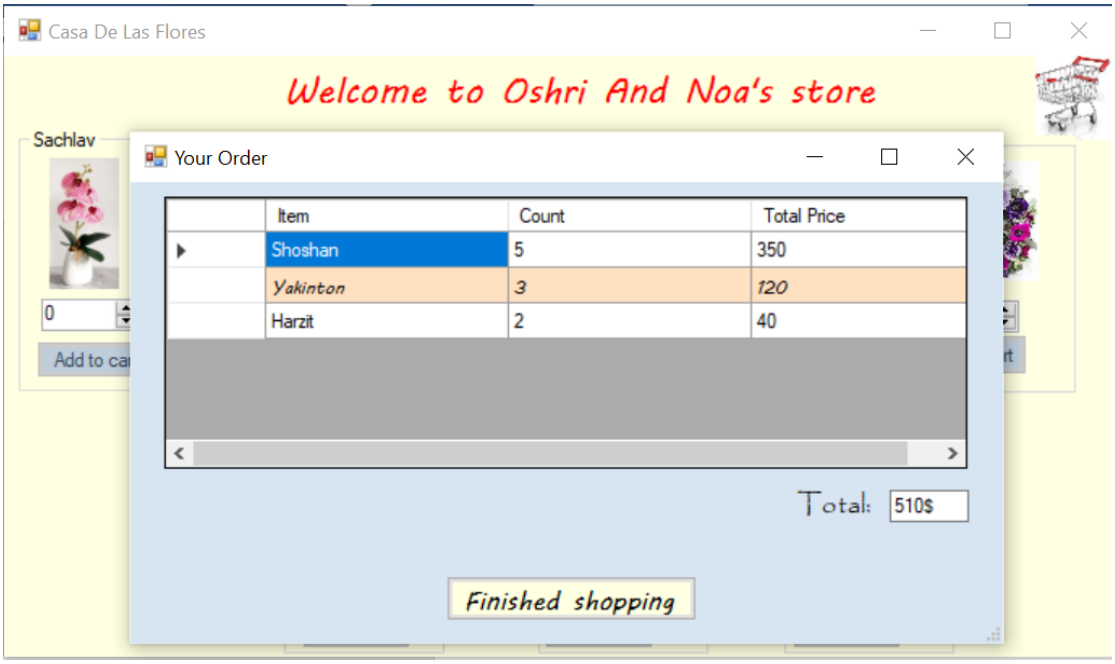


עם לחיצה על פקד ההוספה לסל משתנה צבע הפקד:

בעת עמידה על PICTUREBOX של סל הקניות ישנה הודעה :



בעת לחיצה על PICTUREBOX של סל הקניות נפתח חלון ההזמנה :



בעת לחיצה על ה- BUTTON "Finished shopping" מתקבל MessageBox אודות קליטת ההזמנה: