Week 6 – data structures- Queue applications

Part1- Basics of Queue

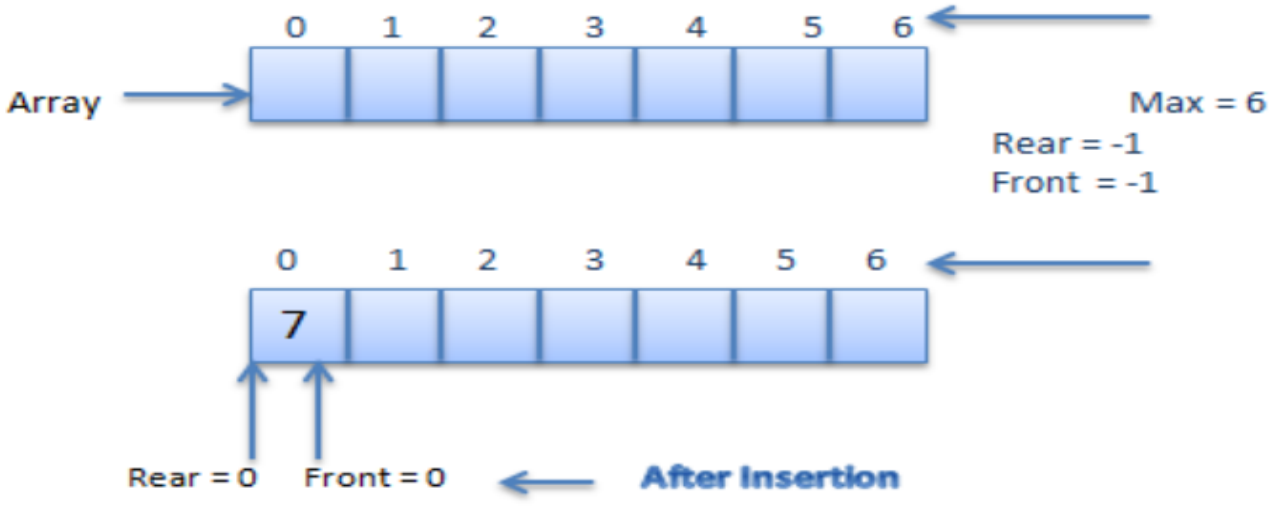
Queues[1]

Queues is a kind of abstract data type where items are inserted one end (rear end) known as **enqueue** operation and deteted from the other end(front end) known as **dequeue** operation.
This makes the queue a **First-In-First-Out (FIFO)** data structure.
The queue performs the function of a buffer.

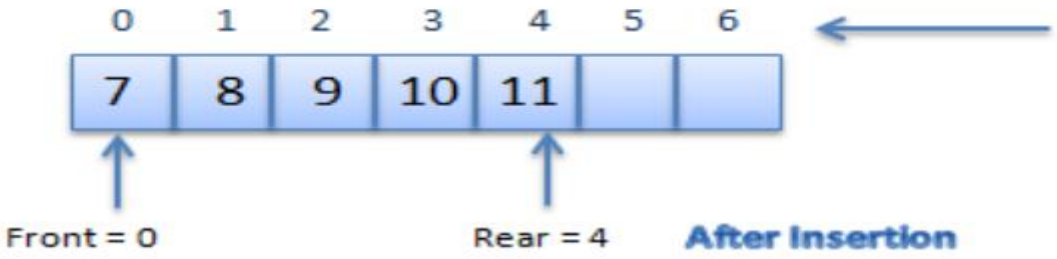
Operation on Queues

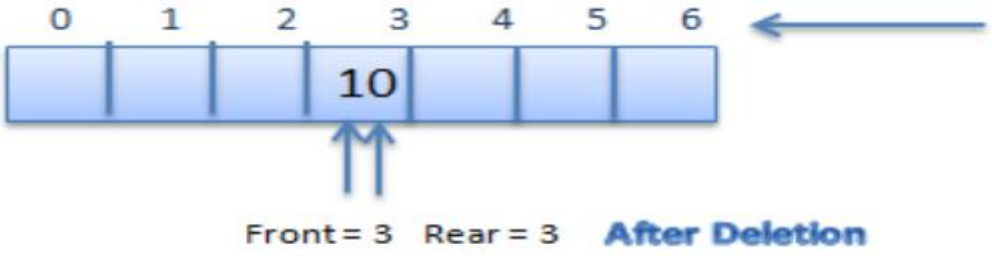
Operation	Description
initialize()	Initializes a queue by adding the value of rear and font to -1.
enqueue()	Insert an element at the rear end of the queue.
dequeue()	Deletes the front element and return the same.
empty()	It returns true(1) if the queue is empty and return false(0) if the queue is not empty.
full()	It returns true(1) if the queue is full and return false(0) if the queue is not full.

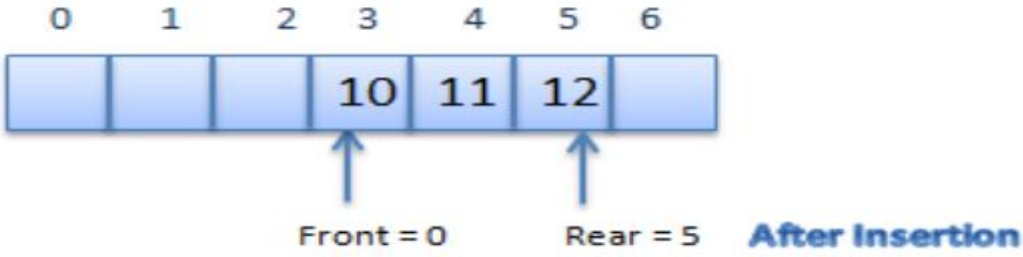
Examples of Queue (Array based)



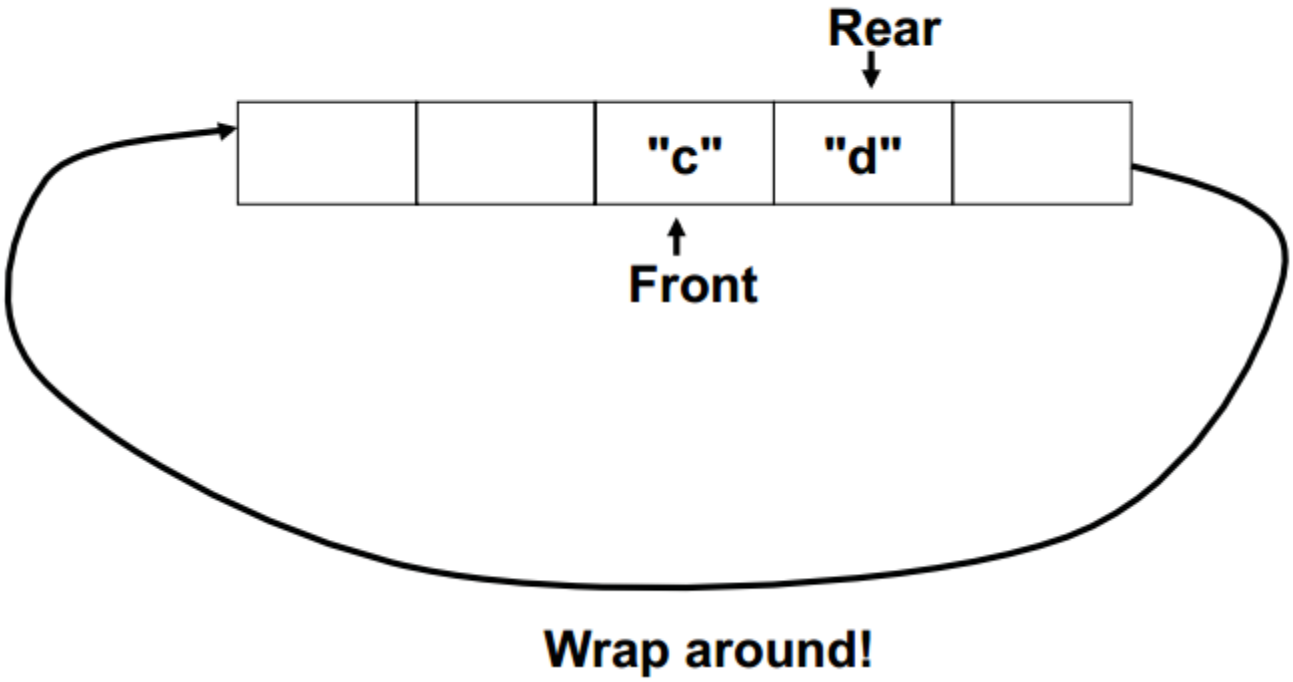
[1] http://scanftree.com/Data_Structure/Queues







Circular array and linked list[2]



[2] http://ocw.mit.edu/courses/civil-and-environmental-engineering/1-00-introduction-to-computers-and-engineering-problem-solving-spring-2012/lecture-notes/MIT1_00S12_Lec_35.pdf

Part2- Problems and Activities on Queues and stacks

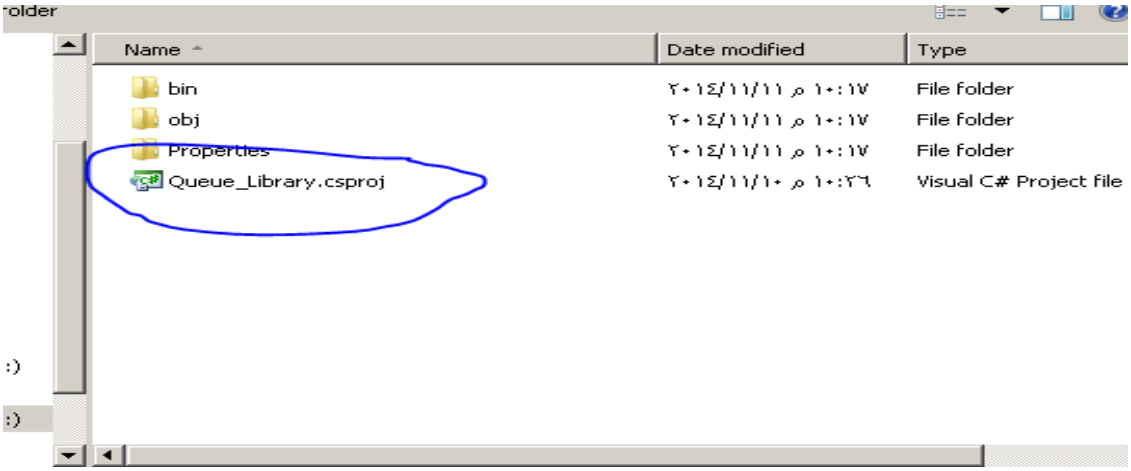
في هذا الجزء نعرض بعض المشكلات التي يمكن استخدام queues and stacks في حلها، هذه المشاكل قد تحتاج لبعض التفكير في كيفية الاستفادة من queue and stackفيما يناسب المشكلة المعروضة، ومن أجل هذا قمنا بعمل مكتبة library يمكن استخدامها في أكوادك للتعامل مباشرة مع queue, circular linked list, and stacks

Activity 0: importing the course library for queues and stack

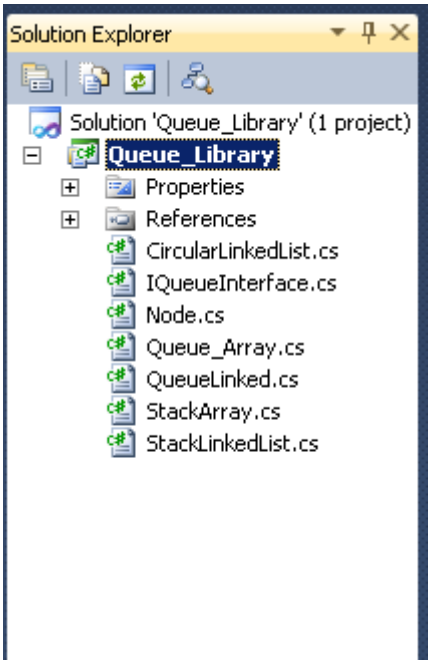
في هذا التمرين نستعرض المكتبة التي يمكنك استخدامها في مشروعات هذا الأسبوع للتعامل مع queue and stacks

Steps:

1- Get the course library of queue and stacks, either from labs CD or by downloading it from:
   a. http://1drv.ms/10SfXMU
   b. or https://drive.google.com/file/d/0Bwsm7JnEnmZwaWpaTFJESnRadXc/view?usp=sharing
2- open visual studio and choose file>open>project/website
3- go to the downloaded folder and choose the project file



4- now you can find the project opened in solution explorer



5- explore the classes in the project,

وهذا ملخص بأدوار هذه classes

Class name	Functionality
Node	The single node in linked list
CircularLinkedList	Circular Linked list: وفيها يكون next لأخر عنصر يشير على أول عنصر، وبالتالي تكون أشبه بدائرة أو حلقة
IQueueInterface	Abstract ADT definitions for the methods in any Queue (array or list)

StackArray	Stack that uses array (from week 5)
StackLinkedList	Stack that uses Linked List (from week 5)
Queue_Array	Queue that uses array
QueueLinked	Queue that uses linked list

6- now, you can add any project to this solution and use the library

Activity 1- Reversing a queue[3]

Problem: use a stack to reverse the order of items in a queue

باستخدام stack اعكس ترتيب العناصر الموجودة في queue فمثلاً إذا كانت العناصر 1-2-3 المطلوب تحويلها ل 3-2-1

Idea:

الفكرة تتلخص في عمل dequeueلكل عنصر في queue واضافته في stack by push، وبالتالي تقوم stack بعمل انعكاس للترتيب لأنها LIFO

Steps:

1- use the course class library in activity 0
2- make new console project
3- add reference to the class library like previous week in lab
4- the code:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using Queue_Stack_Library;//added to solution

namespace Activity_Reverse
{
    class Program
    {
        /*
         * problem:
         * http://pages.cs.wisc.edu/~vernon/cs367/notes/5.STACKS-AND-QUEUES.html#youtry6
         * use stack to reverse the contents of a queue
         */
        static void Main(string[] args)
        {
            Queue_Array<int> numbers = new Queue_Array<int>(5);
            numbers.Enqueue(1);
            numbers.Enqueue(2);
            numbers.Enqueue(3);
            numbers.Enqueue(4);
            numbers.Enqueue(5);

            reverseQueue(numbers);

            while (!numbers.IsEmpty())
            {
                int temp = numbers.Dequeue();
                Console.WriteLine(temp);
            }
            Console.ReadKey();
        }
        static void reverseQueue(Queue_Array<int> numbersQueue)
         {
```

[3] http://pages.cs.wisc.edu/~vernon/cs367/notes/5.STACKS-AND-QUEUES.html#youtry6

```
            StackArray<int> reverseStack = new
StackArray<int>(numbersQueue.GetSize());

            //use stack to store items in rerverse
            while (!numbersQueue.IsEmpty())
            {
                int x = numbersQueue.Dequeue();
                reverseStack.Push(x);
            }

            while (!reverseStack.IsEmpty())
            {
                int x = reverseStack.Pop();
                numbersQueue.Enqueue(x);
            }
        }
    }
}
```

Activity 2: palindrome testing[4]

Problem:

Use a stack and queue to test palindrome. A palindrome is a word, phrase, number, or other sequence of characters which reads the same backward or forward. Famous examples include :

- "A man, a plan, a canal, Panama!",
- "race car",
- "taco cat"
- ""
- "a"
- "aa"
- "aaa"
- "aba"
- "abba"

ال palindrome هي كلمة متناظرة، بمعنى أنها تقرأ من اليمين لليسار مثل قرائتها من اليسار لليمين

Idea:

هناك عدة طرق لفحص كلمة والتأكد من كونها palindrome، الطريقة التي نتبعها اليوم تقوم بوضع حروف الكلمة مرة في queue(بنفس ترتيبها الأصلي) ومرة في stack (بترتيب معكوس) ثم مقارنة الترتيبين، فاذا اختلف أي حرف خلال المقارنة تكون الجملة Not Palindrome، اما اذا تطابقا فتتحقق خاصية palindrome

Code:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using Queue_Stack_Library;

namespace Activity2_Palindrome
{
    class Program
```

[4] http://w3.cs.jmu.edu/lam2mo/cs240_2014_08/lab07-stacks_queues.html and https://en.wikipedia.org/wiki/Palindrome

```
    {
        /*
         * problem:
         * http://w3.cs.jmu.edu/lam2mo/cs240_2014_08/lab07-stacks_queues.html
         * https://en.wikipedia.org/wiki/Palindrome
         * Use a stack and queue to test palindrome,
         * A palindrome is a word, phrase, number, or other sequence of
         * characters which reads the same backward or forward.
         * Allowances may be made for adjustments to capital letters, punctuation, and word
dividers.
         * Famous examples include "A man, a plan, a canal, Panama!", "Amor, Roma", "race car",
"taco cat"
         * Examples of valid palindromes:
           ""
           "a"
           "aa"
           "aaa"
           "aba"
           "abba"
         */
        static void Main(string[] args)
        {
            string p = "race car";
            bool result = testPalindrom(p);
            Console.WriteLine(result);
            Console.ReadKey();
        }
        static bool testPalindrom(string phrase)
        {
            phrase = phrase.ToLower();
            StackLinkedList<char> c_stack = new StackLinkedList<char>();
            QueueLinked<char> c_queue = new QueueLinked<char>();

            for (int i = 0; i < phrase.Length; i++)
            {
                char ch = phrase[i];
                if (ch == ' ')
                    continue;//skip spaces
                c_stack.Push(ch);
                c_queue.Enqueue(ch);
            }

            while (!c_queue.IsEmpty())//same size with stack
            {
                char ch1 = c_queue.Dequeue();
                char ch2 = c_stack.Pop();
                if (ch1 != ch2)
                    return false;
            }

            //no return false happened, so the phrase is palindrome
            return true;

        }
    }
}
```

Activity 3: (Optional) Simulation of printer[5]

Problem:

Simulate a printer that prints several documents, each document has a name and duration of printing. The printer has the following states:

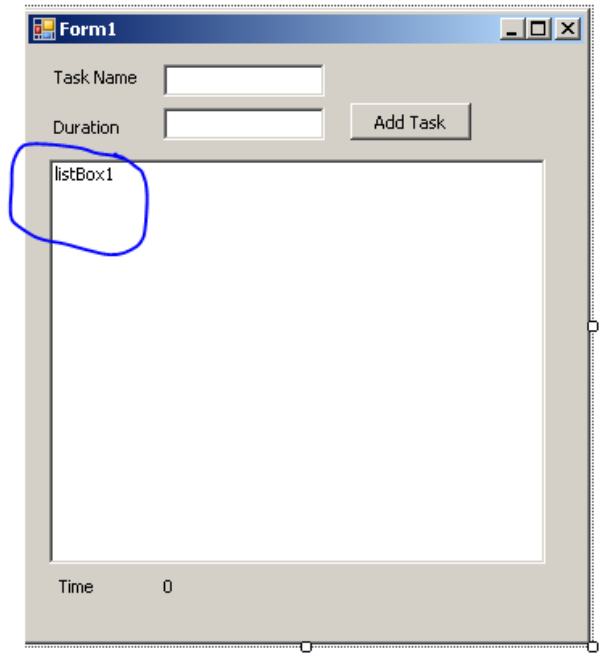
[5] http://interactivepython.org/courselib/static/pythonds/BasicDS/queues.html

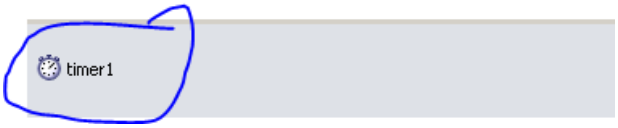
1- The printer has no documents to print
2- The printer has 1 document to print (simple case)
3- The printer is still printing a document, and another document(s) need (s) printing also. In this case put the new document(s) in printer queue till you finish the current document and pop the other document(s).

محاكاة لفكرة الطابعة التي قد يأتها طلب طباعة لملفين في نفس الوقت، فتبدأ في طباعة احدهما وتضع الأخر في queueلحين الانتهاء من الملف الأول، وقد تأتي ملفات أخرى في أثناء هذه العملية، فبالتالي لابد من وضعهم جميعاً في queueلحين الوصول لدورهم في الطباعة، في هذا البرنامج لن نقوم بعمل طباعة فعلية ولكن سنقوم بعمل محاكاة، سيكون لكل ملف مدة زمنية في الطباعة، ويقوم البرنامج باستخدام Timer لتشغيل خطوة في المحاكاة كل ثانية، وفي كل ثانية تمر تقل المدة المتبقية في الملف الجاري طباعته، حتى نصل ل 0 وعندها يتم الانتها من الملف واسترجاع الملف التالي من queue

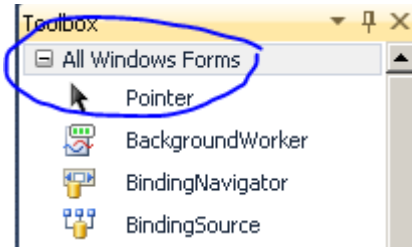
Steps:

1- add new windows application to the solution
2- use the course queue library as a reference to your windows application
3- add on the form: timer, listbox, and some buttons, labels and textboxes like the following

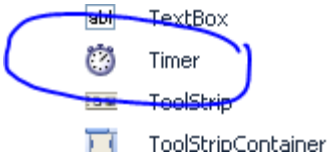




Note: to get timer from toolbox, open the "all windows forms" tab and select timer

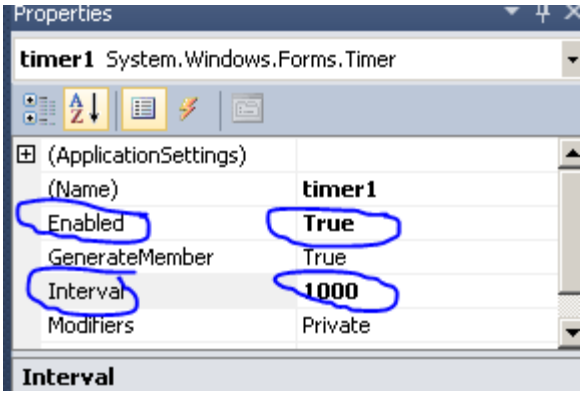


And scroll down to find the timer, when you add it to form it doesn't appear on form itself but displayed in the bottom of visual studio



4- the listbox will display the printer message

5- right click the timer and choose properties and set the following



6- double click the timer, button add to generate their events methods

7- the code:

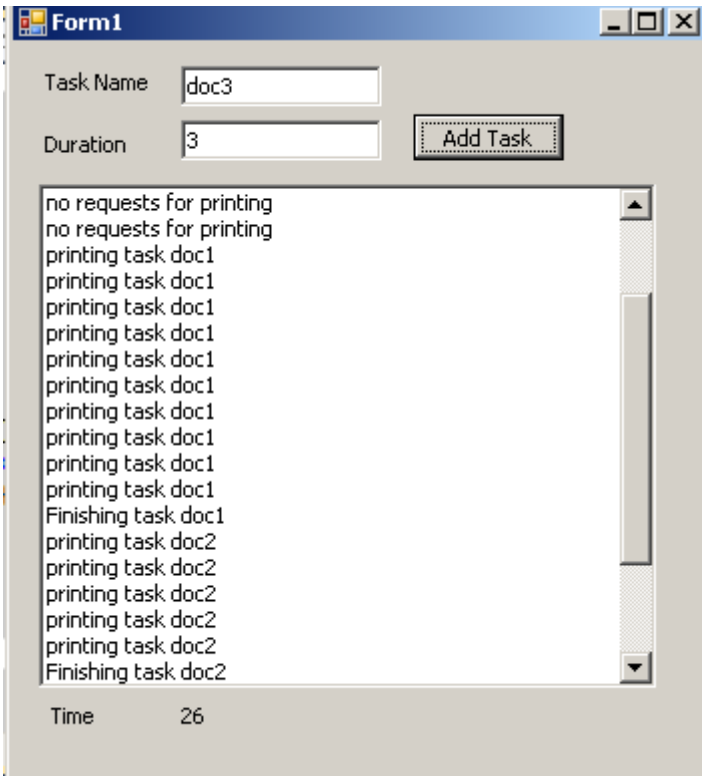
```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;
using Queue_Stack_Library;

namespace Activity3_PrinterSimulationGUI
{
    /*
    * problem:
    * http://interactivepython.org/courselib/static/pythonds/BasicDS/queues.html
    * simulation of printer queue of files to print
    */
    public partial class Form1 : Form
    {
        QueueLinked<PrintingTask> tasksQueue = new QueueLinked<PrintingTask>();
        int time = 0;
        public Form1()
        {
            InitializeComponent();
        }

        private void timer1_Tick(object sender, EventArgs e)
        {
            //display time clock
            label_time.Text = time.ToString();
            time++;
            simulatePrinter();

        }
        void simulatePrinter()
        {

            if (tasksQueue.IsEmpty())
                listBox1.Items.Add( "no requests for printing");
            else
            {
                //peek first
                PrintingTask currentTop = tasksQueue.GetFront();
                currentTop.duration--;//decrease time
                listBox1.Items.Add("printing task "+ currentTop.Name);

                if (currentTop.duration == 0)//remove task
                {
                    tasksQueue.Dequeue();
                    listBox1.Items.Add("Finishing task " + currentTop.Name);
```

```
            }
        }
    }

    private void button1_Click(object sender, EventArgs e)
    {
        int dur = int.Parse(textBox_duration.Text);
        PrintingTask task = new PrintingTask(textBox_name.Text, dur);

        //add new task to queue, avoid read write problems
        timer1.Enabled = false;//why? to avoid threading problems on queue
        tasksQueue.Enqueue(task);
        timer1.Enabled = true;
    }
}
class PrintingTask
{
    /// <summary>
    /// task name
    /// </summary>
    public string Name;

    /// <summary>
    /// how long it's remaining
    /// </summary>
    public int duration;

    public PrintingTask(string n, int d)
    {
        Name = n;
        duration = d;
    }
}
}
```

8- run the project and add the following documents:

   a. doc1 (10 seconds)
   b. doc2 (5 seconds)
   c. doc3(2 seconds)

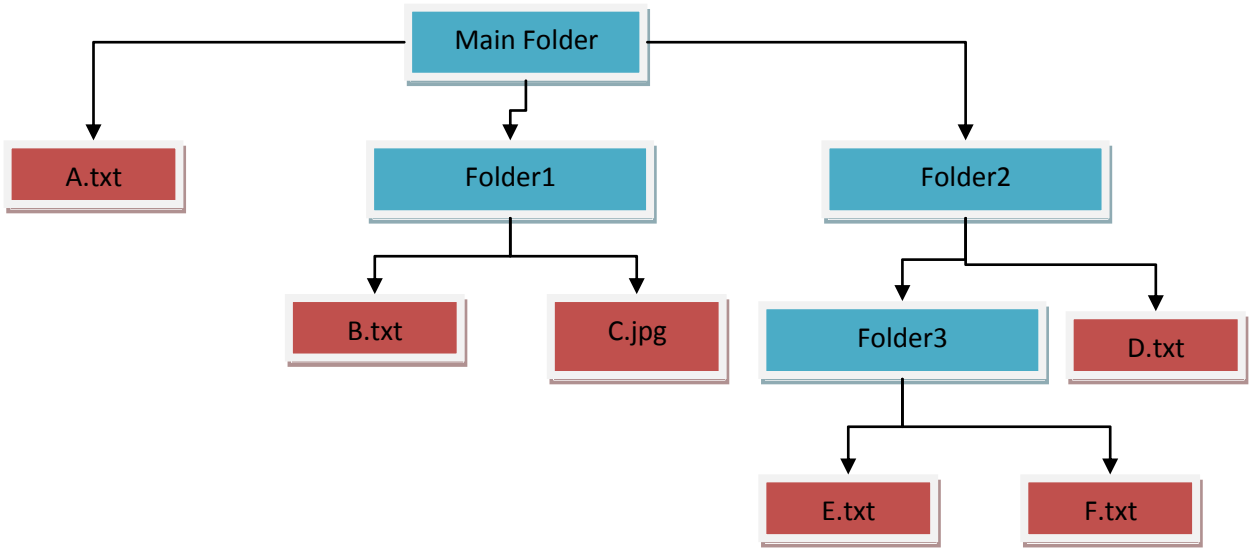
9- you can get something like:



Activity 4: Flatten Folder and its sub-folders:[6]

Problem:

Get all files in a certain folder including all the files in the sub-folders



في هذا البرنامج نقوم بالحصول على كل الملفات (الملونة في الرسم باللون الأحمر) داخل مجلد معين وكل المجلدات الفرعية داخله (المجلدات باللون الأزرق) ، فمثلاً عندما يعمل هذا البرنامج على المجلد Main Folderالموجود في الرسم فالمفروض أن يكون الناتج اسماء الملفات فقط :

A.txt, B.txt, C.jpg, D.txt, E.txt, F.txt

بغض النظر عن كونها داخل main folderمباشرة أو داخل مجلد أخر داخل main folder ولنقم بتقديم مثال بسيط لعمل tracingلهذا البرنامج

	Operation	Input	Folders Queue	Files Queue
1	Initial	Main folder	MainFolder	---
2	-Pop **Main Folder** from Folders Queue -Get files in MainFolder	A.txt	----	A.txt (the file inside Main Forlder)
3	-Get sub-folders of Main Folder and add to Folders Queue	Folder1, Folder2	Folder1 Folder2	A.txt
4	-Pop Folder1 -Get files of Folder1	B.txt, C.jpg	Folder2	A.txt B.txt C.jpg
5	Pop Folder2 -Get files of Folder2	D.txt	---	A.txt B.txt C.jpg D.txt
6	Get sub-folders of Folder2	Folder3	Folder3	A.txt B.txt C.jpg D.txt
7	-Pop folder3 -get folder 3 files	E.txt F.txt	---	A.txt B.txt C.jpg D.txt E.txt F.txt

[6] http://stackoverflow.com/questions/2106877/is-there-a-faster-way-than-this-to-find-all-the-files-in-a-directory-and-all-sub

FileInfo[7] gets file statistics. It retrieves information about a specific file or directory from the file system in a C# program. The FileInfo type provides a host of methods and properties that helps you detect the status of the file.

Attributes

Every file on the Windows File system stores a set of attributes that tell you certain things about the file. You can detect its visibility, whether it is a directory, and if it is read-only.

Length

How many bytes are in a file? The Length property on the FileInfo type provides a way to effectively determine this.
It returns a figure in bytes,
not megabytes
or kilobytes. You may need to convert the value.

Program that uses Length property: C#

```
using System;
using System.IO;

class Program
{
    static void Main()
    {
        FileInfo info = new FileInfo("C:\\a");
        long value = info.Length;
        Console.WriteLine(value);
    }
}
```

DirectoryInfo[8]

The DirectoryInfo is another way of accessing the Directory type functionality. We access important properties and methods on a DirectoryInfo. We create DirectoryInfo by passing a directory path to its constructor. It is a class.

Program that uses DirectoryInfo: C#

```
using System;
using System.IO;

class Program
{
    static void Main()
    {
        // Get info.
        DirectoryInfo info = new DirectoryInfo(@"C:\perls\");

        // Write name.
        Console.WriteLine(info.Name);

        // Write file count.
        FileInfo[] array = info.GetFiles();
```

[7] http://www.dotnetperls.com/fileinfo
[8] http://www.dotnetperls.com/directory

```
        Console.WriteLine(array.Length);
    }
}
```

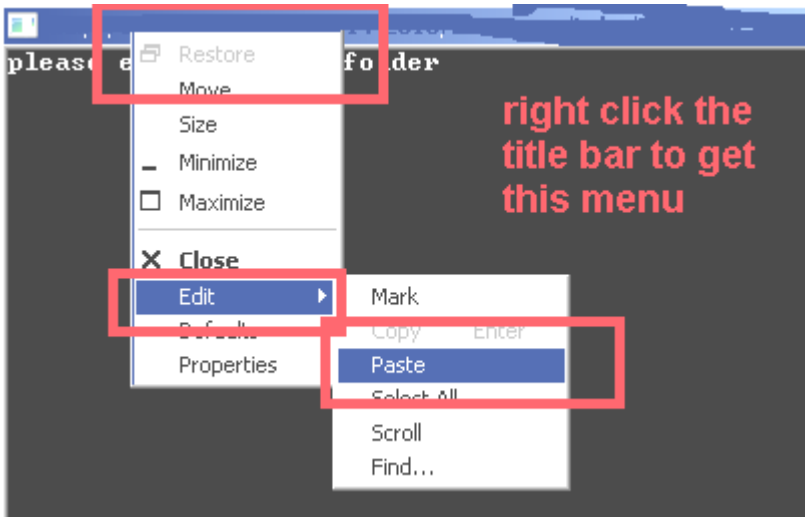
Steps:

1- make new console application
2- add reference to course class library
3- code:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.IO;
using Queue_Stack_Library;

namespace Activity4_Flatten_Folder
{
    class Program
    {
        /* problem
         * http://stackoverflow.com/questions/2106877/is-there-a-faster-way-than-this-to-find-all-the-files-in-a-directory-and-all-sub
         * get all files in a certain folder including all the files in the sub-folders
         */
        static void Main(string[] args)
        {
            Console.WriteLine("please enter path to folder");
            string path = Console.ReadLine();

            DirectoryInfo dir = new DirectoryInfo(path);

            //folders to get their files
            QueueLinked<DirectoryInfo> foldersQueue = new QueueLinked<DirectoryInfo>();

            //found files
            QueueLinked<string> filesQueue = new QueueLinked<string>();

            //start with main directory in folders queue
            foldersQueue.Enqueue(dir);

            //objects to use inside the loop
            DirectoryInfo[] foldersArray;
            FileInfo[] filesArray;
            DirectoryInfo tempDir;

            while (!foldersQueue.IsEmpty())
            {
                //get a folder from queue
                tempDir = foldersQueue.Dequeue();

                //get the current folder files
                filesArray = tempDir.GetFiles();

                //get sub folders
                foldersArray = tempDir.GetDirectories();

                //add files names to files queue
                foreach (FileInfo f in filesArray)
                {
                    filesQueue.Enqueue(f.Name);
                }

                //add sub-folders to folders queue to get their child folders also later
```

```
            foreach (DirectoryInfo d in foldersArray)
            {
                foldersQueue.Enqueue(d);
            }

        }
        //now this while has ended, so we got all the files from all the sub-folders
        //print them
        Console.WriteLine("Found "+ filesQueue.GetSize() +" files:");
        while (!filesQueue.IsEmpty())
        {
            Console.WriteLine(filesQueue.Dequeue());
        }

        Console.ReadKey();

    }
  }
}
```

4- now, make some folders and files like the above drawing of the Main Folder Example, and pass this folder to the application at running, you can copy and paste the folder path



Self-exercise (4 degrees):

Modify Activity 4 to calculate :

1- the total size of the main folder (Hint: use FileInfo.length)
2- get the largest file name from the flattened file list and print its name