SIT221 –DATA STRUCTURES AND ALGORITHMS

LAB9: GRAPHS

LAB OBJECTIVE:

The objective of this lab is to learn how to use QuickGraph to model a graph problem like ShoppingCenter POIs.

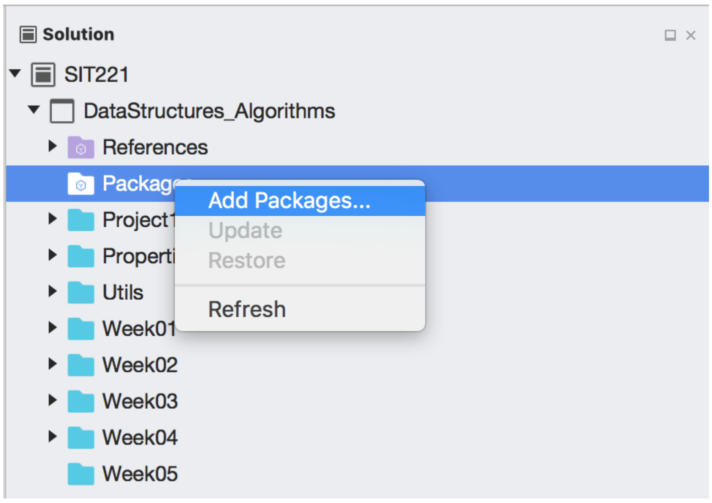
SUBMISSION INSTRUCTIONS

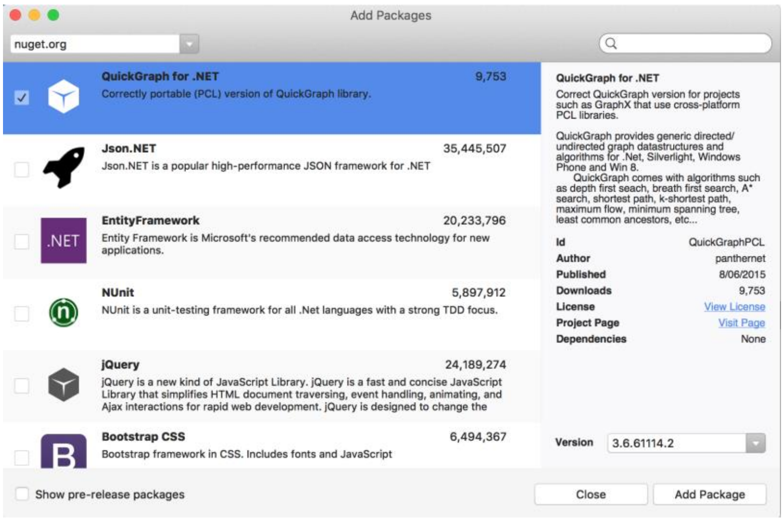
Please submit your work to Week09 assignment folder

PREPARATION

1. Download the template project available in week09 resources folder. The solution has two projects: **DataStructures_Algorithms & Runner** projects.
2. Check the QuickGraph library here: https://quickgraph.codeplex.com/documentation
3. Add QuickGraph Package to your project (if it is not already installed). How? See lecture slides or as below:

In Xamarin Studio





☑ **QuickGraph for .NET** 9,753
Correctly portable (PCL) version of QuickGraph library.

□ **Json.NET** 35,445,507
Json.NET is a popular high-performance JSON framework for .NET

□ **EntityFramework** 20,233,796
Entity Framework is Microsoft's recommended data access technology for new applications.

□ **NUnit** 5,897,912
NUnit is a unit-testing framework for all .Net languages with a strong TDD focus.

□ **jQuery** 24,189,274
jQuery is a new kind of JavaScript Library. jQuery is a fast and concise JavaScript Library that simplifies HTML document traversing, event handling, animating, and Ajax interactions for rapid web development. jQuery is designed to change the

□ **Bootstrap CSS** 6,494,367
Bootstrap framework in CSS. Includes fonts and JavaScript

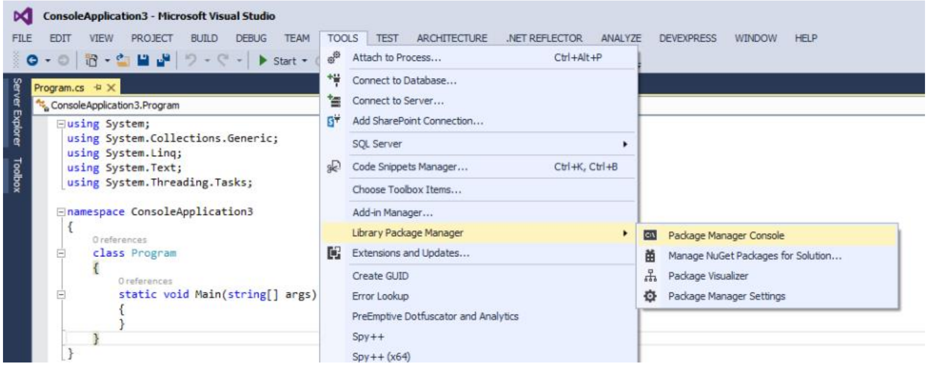
QuickGraph for .NET
Correct QuickGraph version for projects such as GraphX that use cross-platform PCL libraries.

QuickGraph provides generic directed/undirected graph datastructures and algorithms for .Net, Silverlight, Windows Phone and Win 8.
QuickGraph comes with algorithms such as depth first seach, breath first search, A* search, shortest path, k-shortest path, maximum flow, minimum spanning tree, least common ancestors, etc...

Id	QuickGraphPCL
Author	panthernet
Published	8/06/2015
Downloads	9,753
License	View License
Project Page	Visit Page
Dependencies	None

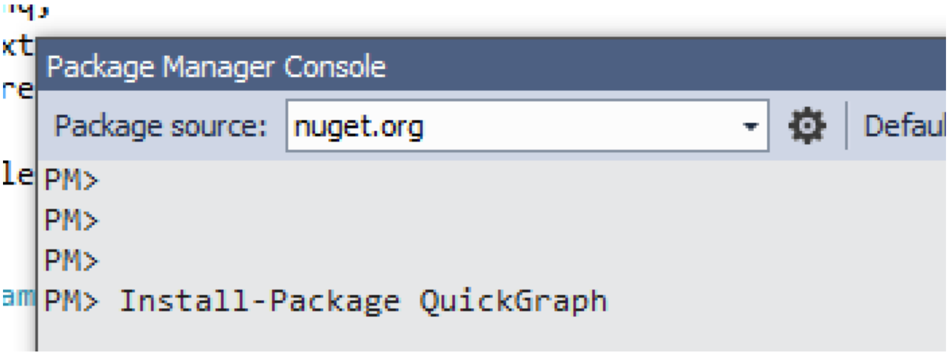
Version 3.6.61114.2

OR – In Visual Studio



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

namespace ConsoleApplication3
{
    0 references
    class Program
    {
        0 references
        static void Main(string[] args)
        {
        }
    }
}
```



```
PM>
PM>
PM>
PM> Install-Package QuickGraph
```

LAB TASKS

1. SHOPPING CENTER GRAPH

In this task we want to use BidirectionalGraph data structure to store routes between PointOfInterests in a shoppingcenter and be able to find shortest path between two Points of interest in the shopping center.

1. There is a class called **PointOfInterest** in **Week07**. This class maintains information about a POI including Name, Description, Location Details, and List of items or services.
2. There is a class called **GraphEdge** in **Week09.** This that is used to store information about the route between two point of interests POI1 and POI2 including source node (POI 1), target node (POI 2), Description, and Distance between both points.
3. We want to create a class called **ShoppingCenterGraph** that has the following data and methods
    a. A property & field of type BidirectionalGraph, call it **POIsGraph**. The graph nodes will be of type **PointOfInterest** , and the graph edges will be of type **GraphEdge<PointOfInterest>**.
    b. **Constructor**: In the constructor please initialize your **POIsGraph** and add at least four POIs with edges between them.
    c. **public List<PointOfInterest> FindDirectPOIsFrom(string POIName)**: This method should return all possible POIs **directly** reachable from the POI with the given POIName (parameter value).

       How? Your method should loop on all graph nodes, and check if a given node/POI has a Name equal to the input parameter **POIName**. Once this node/POI is found, you then need to get the OutEdges of this node, loop on them while extracting the Target node of each edge – e.g. edge.Target, add this Target node to the output list. Finally return this list to the user.

4. In the Runner09_Task1 class / Run method please do the following:
    a. Create an object of the **ShoppingCenterGraph – call it myShoppingCenterGraph.**
    b. Call the **myShoppingCenterGraph**.FindDirectPOIsFrom(VALUE??) and display the returned list of POIs on the screen.