تمرین ۷ درس ساختمان داده

مریم سادات هاشمی
سید صالح اعتمادی

دانشگاه علم و صنعت ۹۸_۹۷

لطفا به نکات زیر توجه کنید:

- مهلت ارسال این تمرین شنبه ۲۶ آبان ماه ساعت ۱۱:۵۹ ب.ظ است.
- این تمرین شامل سوال های برنامه نویسی می باشد، بنابراین توجه کنید که حتماً موارد خواسته شده را رعایت کنید. .
- نام شاخه، پوشه و پول ریکوست همگی دقیقا "A۷" باشد.
- در صورتی که به اطلاعات بیشتری نیاز دارید می توانید با ایدی تلگرام @maryam_sadat_hashemi در ارتباط باشید.
- اگر در حل تمرین شماره ی ۷ مشکلی داشته اید،لطفا به `https://calendly.com/hashemi-maryam-sadat` مراجعه کنید و زمانی را برای رفع اشکال تنظیم کنید.

موفق باشید.

توضیحات کلی تمرین

تمرین این هفته ی شما، ۳ سوال دارد که باید به همه ی این سوال ها پاسخ دهید. برای حل این سری از تمرین ها مراحل زیر را انجام دهید:

۱. ابتدا مانند تمرین های قبل، یک پروژه به نام A۷ بسازید.

۲. کلاس هر سوال را به پروژه ی خود اضافه کنید و در قسمت مربوطه کد خود را بنویسید. هر کلاس شامل دو متد اصلی است:

متد اول: تابع solve است که شما باید الگوریتم خود را برای حل سوال در این متد پیاده سازی کنید.

متد دوم: تابع process است که مانند تمرین های قبلی در TestCommon پیاده سازی شده است. بنابراین با خیال راحت سوال را حل کنید و نگران تابع process نباشید! زیرا تمامی پیاده سازی ها برای شما انجام شده است و نیازی نیست که شما کدی برای آن بزنید.

۳. اگر برای حل سوالی نیاز به تابع های کمکی دارید؛ می توانید در کلاس مربوط به همان سوال تابع تان را اضافه کنید.

اکنون که پیاده سازی شما به پایان رسیده است، نوبت به تست برنامه می رسد. مراحل زیر را انجام دهید.

۱. یک UnitTest برای پروژه ی خود بسازید.

۲. فولدر TestData که در ضمیمه همین فایل قرار دارد را به پروژه ی تست خود اضافه کنید.

۳. فایل GradedTests.cs را به پروژه ی تستی که ساخته اید اضافه کنید. توجه کنید که مانند تمرین های قبل، لازم نیست که برای هر سوال TestMethod بنویسید. تمامی آنچه که برای تست هر سوالتان نیاز دارید از قبل در این فایل برای شما پیاده سازی شده است.

```
using Microsoft.VisualStudio.TestTools.UnitTesting;
using A7;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using TestCommon;

namespace A7.Tests
{
    [TestClass()]
    – references | Sauleh Eetemadi, 9 days ago | 1 author, 1 change
    public class GradedTests
    {
        [TestMethod(), Timeout(1000)]
        [DeploymentItem("TestData", "A7_TestData")]
        – references | Sauleh Eetemadi, 9 days ago | 1 author, 1 change
        public void SolveTest()
        {
            Processor[] problems = new Processor[] {
                new MaximumGold("TD1"),
                new PartitioningSouvenirs("TD2"),
                new MaximizingArithmeticExpression("TD3")
            };

            foreach (var p in problems)
            {
                TestTools.RunLocalTest("A7", p.Process, p.TestDataName);
            }
        }
    }
}
```

دقت کنید که TestCommon تغییر یافته است. بنابراین شما باید نسخه ی جدید آن را با دستورgit Pull دریافت کنید .

۱ Maximum Amount of Gold

فرض کنید مجموعه ای از شمش های طلا را به شما داده ایم و شما باید تا جایی که امکان دارد شمش های طلا را در کیفتان قرار دهید. فقط یک کپی از هر شمش وجود دارد و هر شمش را شما می توانید بردارید یا نه (بنابراین شما نمی توانید کسری از شمش را بردارید).
اگر n تا شمش طلا داشته باشیم با Dynamic Programming بیشترین وزن ممکن از طلا ها که در کیفی به ظرفیت w جا می شود، را پیدا کنید.
مطابق شکل زیر، شما باید الگوریتم خود را در تابع Solve که در کلاس MaximumGold قرار دارد، بنویسید.

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

namespace A7
{
    1 reference | Sauleh Eetemadi, 9 days ago | 1 author, 1 change
    public class MaximumGold : Processor
    {
        0 references | Sauleh Eetemadi, 9 days ago | 1 author, 1 change
        public MaximumGold(string testDataName) : base(testDataName) { }

        3 references | Sauleh Eetemadi, 9 days ago | 1 author, 1 change
        public override string Process(string inStr) =>
            TestTools.Process(inStr, (Func<long, long[], long>)Solve);

        1 reference | Sauleh Eetemadi, 9 days ago | 1 author, 1 change
        public long Solve(long W, long[] goldBars)
        {
            return 0;
        }
    }
}
```

۲ Partitioning Souvenirs

شما و دو نفر از دوستانتان پس از بازدید از کشورهای مختلف، به خانه بازگشته اید. حالا شما می خواهید تمام سوغاتی هایی که سه نفری خریداری کرده اید را به طور مساوی بین همدیگر تقسیم کنید. در فایل ورودی این سوال، در خط اول تعداد سوغاتی ها و در خط بعدی مقدار سوغاتی ها خواهد بود. اگر سوغاتی ها را می توان به صورت مساوی تقسیم کرد در فایل خروجی عدد ۱ و در غیر این صورت عدد صفر خواهد بود. به مثال های زیر توجه کنید.

ورودی :

4

$3, 3, 3, 3$

خروجی :

0

ورودی:

11

$17, 59, 34, 57, 17, 23, 67, 1, 18, 2, 59$

خروجی:

1

$34 + 67 + 17 = 23 + 59 + 1 + 17 + 18 = 59 + 2 + 57.$

مطابق شکل زیر، شما باید الگوریتم خود را در تابع Solve که در کلاس Partitioning–Souvenirs قرار دارد، بنویسید.

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

namespace A7
{
    1 reference | Sauleh Eetemadi, 9 days ago | 1 author, 1 change
    public class PartitioningSouvenirs : Processor
    {
        0 references | Sauleh Eetemadi, 9 days ago | 1 author, 1 change
        public PartitioningSouvenirs(string testDataName) :
            base(testDataName) { }

        3 references | Sauleh Eetemadi, 9 days ago | 1 author, 1 change
        public override string Process(string inStr) =>
            TestTools.Process(inStr, (Func<long, long[], long>)Solve);

        1 reference | Sauleh Eetemadi, 9 days ago | 1 author, 1 change
        public long Solve(long souvenirsCount, long[] souvenirs)
        {
            return 0;
        }
    }
}
```

۳ Maximum Value of an Arithmetic Expression

در این سوال، شما باید با روش Dynamic Programing به گونه ای پرانتز ها را به یک عبارت ریاضی اضافه کنید که مقدار عبارت ریاضی به حداکثر برسد. خط اول فایل ورودی شامل یک رشته s با طول $2n+1$ است که n از یک تا ۱۴ می باشد. رشته را می توانیم به صورت زیر نمایش دهیم:

$$s_0, s_1, s_2, ..., s_n$$

کاراکتر موجود در موقعیت های زوج رشته ی s یک رقم می باشد و کاراکتر موجود در موقعیت های فرد رشته ی s سه عمل +، ـ و * می باشد.
مطابق شکل زیر، شما باید الگوریتم خود را در تابع Solve که در کلاس Maximizin–gArithmeticExpression قرار دارد، بنویسید.

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

namespace A7
{
    1 reference | Sauleh Eetemadi, 9 days ago | 1 author, 1 change
    public class MaximizingArithmeticExpression : Processor
    {
        0 references | Sauleh Eetemadi, 9 days ago | 1 author, 1 change
        public MaximizingArithmeticExpression(string testDataName) :
            base(testDataName) { }

        3 references | Sauleh Eetemadi, 9 days ago | 1 author, 1 change
        public override string Process(string inStr) =>
            TestTools.Process(inStr, (Func<string, long>)Solve);

        1 reference | Sauleh Eetemadi, 9 days ago | 1 author, 1 change
        public long Solve(string expression)
        {
            return 0;
        }
    }
}
```