تمرین ۵ 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 ۵ 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 تابع ها و تست ها باید همگی مطابق آنچه که خواسته شده است؛ باشد.
- نام شاخه، پوشه و پول ریکوست همگی دقیقا "A۵" 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 ۴ 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،۶ سوال دارد که باید به همه ی این سوال ها پاسخ دهید.
ابتدا مانند تمرین های قبل، یک پروژه به نام A۵ بسازید. شما باید برای هر سوال یک تابع بسازید و الگوریتم خود را داخل آن تابع پیاده سازی کنید. دقت کنید که در هر بخش توضیح دادیم که اسم تابع ها را چگونه انتخاب کنید.
بعد از اینکه الگوریتم خود را در تابع مورد نظر پیاده سازی کردید؛ برای بررسی درستی الگوریتم، شما باید الگوریتم خود را تست کنید. بنابراین شما نیاز دارید یک Unit Test برای پروژه ی خود بسازید. سپس باید فولدر TestData که در ضمیمه همین فایل قرار دارد را به پروژه ی تست خود اضافه کنید. داخل فولدر TestData شش فولدر دیگر قرار دارد که در هر کدام testcase های هر سوال قرار داده شده است. برای مثال testcase های سوال یک در فولدرTD۱ می باشد.
بعد از اینکه فولدر TestData را به پروژه ی خود اضافه کردید؛ باید برای هر الگوریتم خود یک TestMethod نیز بنویسید. (دقت کنید که در توضحات هر سوال نام TestMethod را هم مشخص کردیم و TestMethod های شما هم دقیقا باید به همین اسامی باشند.) برای نوشتن تست هم از TestCommon (مثل تمرین های قبل) باید استفاده کنید. دقت کنید که تغییر کوچکی در TestCommon برای این تمرین انجام شده است. بنابراین شما باید ابتدا نسخه ی جدید این کلاس را با دستورgit Pull دریافت کنید . در این تمرین تابع process را در TestCommon برای شما پیاده سازی کرده ایم و تنها کافی است برای تابع process هر سوال از تابع process موجود در Deligate TestCommon، کنید. برای مثال برای سوال اول به صورت زیر می شود. همین کار را برای سایر سوال ها انجام دهید.
به شکل زیر دقت کنید.

```
public static string ProcessBinarySearch1(string inStr) =>
    TestTools.Process(inStr, BinarySearch1);
```

متد RunLocalTest سه ورودی می گیرد:
ورودی اول: نام assignment است. بنابراین برای این تمرین، این پارامتر A۵ خواهد بود.
ورودی دوم: یک تابع است که ورودی و خروجی این تابع از نوع string است.بنابراین شما باید یک تابع process هم برای هر سوال در پروژه ی اصلی خود داشته باشید که به عنوان ورودی به متد RunLocalTest بدهید. هدف از تابع process این است که داده ی مورد نظر را از ورودی می گیرد و سپس پردازش لازم روی دیتای ورودی انجام شود تا ورودی مطابق با نوع ورودی تابع الگوریتم شما باشد. برای مثال اگر تابع الگوریتم شما داده را به صورت int دریافت می کند شما باید داده را به صورت int در بیاورید تا بتوانید داده را به تابع الگوریتم خود بدهید. و در نهایت هم خروجی تابع process همان خروجی تابع الگوریتم شما خواهد بود. دقت کنید که در توضحات هر سوال، نام تابع Process را هم مشخص کردیم و تابع های شما هم دقیقا باید به همین اسامی باشند.
ورودی سوم : نام فولدری است که testcase های مربوط به آن سوال قرار دارد مثلا برای سوال یک این ورودی TD۱ خواهد بود و برای سوال دو TD۲ و الی آخر.
اکنون شما موفق شدید تابع الگوریتم و process و TestMethod مربوط به سوال را پیاده سازی کنید و باید الگوریتم خود را تست کنید.

۱ Binary Search

فرض کنید که دو آرایه ی a و b به طول n در اختیار دارید. شما باید بررسی کنید که آیا هر یک از عنصر های آرایه ی b در آرایه a موجود است یا خیر. اگر در آرایه ی a باشد شما باید Index آن را در آرایه ی a به عنوان خروجی برگردانید و اگر در آرایه ی a نباشد عدد ـ۱ را برگردانید. دقت کنید که آرایه ی a به صورت صعودی است و همه ی عناصر آن از یکدیگر متمایز هستند. به مثال زیر توجه کنید:

$a : 1, 5, 8, 12, 13$
$b : 8, 1, 23, 1, 11$

در مثال بالا شما می بینید که ۸ و ۱ در آرایه ی a وجود دارند و Index آن ها به ترتیب ۲ و ۰ می باشد ولی ۱۱ و ۲۳ در a وجود ندارد. بنابراین خروجی به صورت زیر خواهد بود:

$2, 0, -1, 0, -1$

برای حل این سوال شما باید یک الگوریتمconquer and divide بنویسید.

- محدودیت زمانی : **۱۰۰۰ میلی ثانیه**

اسامی توابع شما در این تمرین باید به صورت زیر باشد.

- Algorithm Function : BinarySearch1
- Process Function : ProcessBinarySearch1
- Test Function : Graded_BinarySearch1Test

```
public static long[] BinarySearch1(long[] a , long [] b)
{
    //write your code here
    return new long[] {0};
}
```

```
public static string ProcessBinarySearch1(string inStr) =>
    TestTools.Process(inStr, BinarySearch1);
```

```
[TestMethod(),Timeout(1000)]
[DeploymentItem(@"TestData", "A5_TestData")]
0 references | 0 changes | 0 authors, 0 changes
public void Graded_BinarySearch1Test()
{
    TestTools.RunLocalTest("A5", Program.ProcessBinarySearch1, "TD1");
}
```

۲ Majority Element

فرض کنید که دنباله ای از اعداد به صورت $a_1, a_2, ..., a_n$ داریم. شما باید یک الگوریتم conquer and divide بنویسید که چک کند آیا عنصری در دنباله وجود دارد که بیش از $n/2$ بار تکرار شده باشد یا خیر. اگر چنین عنصری وجود داشته باشد، به آن majority element می گوییم و شما در این حالت بایدد عدد یک را برگردانید در غیر این صورت عدد صفر را برگردانید.

در testcase های این سوال خط اول تعداد المان های دنباله و خطوط بعدی المان های آرایه هستند. و خروجی هم عدد یک یا صفر است. به مثال زیر توجه کنید:

5
2, 3, 9, 2, 2
output: 1

در این دنباله عدد۲، majority element است.

- محدودیت زمانی : **۱۰۰۰ میلی ثانیه**

اسامی توابع شما در این تمرین باید به صورت زیر باشد.

- Algorithm Function : MajorityElement2
- Process Function : ProcessMajorityElement2
- Test Function : Graded_MajorityElement2Test

```
public static long MajorityElement2(long n, long[] a)
{
    //write your code here
    return 0;
}
```

```
public static string ProcessMajorityElement2(string inStr) =>
    TestTools.Process(inStr, (Func<long, long[], long>)MajorityElement2);
```

```
[TestMethod(), Timeout(1000)]
[DeploymentItem(@"TestData", "A5_TestData")]
0 references | 0 changes | 0 authors, 0 changes
public void Graded_MajorityElement2Test()
{
    TestTools.RunLocalTest("A5", Program.ProcessMajorityElement2, "TD2");
}
```

۳ Improving Quick Sort

در کلاس درس شما پیاده سازی الگوریتم Quick Sort را دیدید. اکنون شما باید این الگوریتم را به گونه ای تغییر دهید که برای آرایه هایی که تعداد المان های مساوی زیادی دارند هم، سریع عمل کند. برای راهنمایی شما باید به جای ۲ partition بندی، ۳ partition بندی داشته باشید.

testcase های این تمرین هم به این شکل است که در فایل ورودی خط اول تعدا المان های آرایه و در خطوط بعدی هر یک از المان های آرایه قرار دارد.و در فایل خروجی هم آرایه ی مرتب شده می باشد.

- محدودیت زمانی : **۱۰۰۰ میلی ثانیه**

اسامی توابع شما در این تمرین باید به صورت زیر باشد.

- Algorithm Function : ImprovingQuickSort3
- Process Function : ProcessImprovingQuickSort3
- Test Function : Graded_ImprovingQuickSort3Test

```
public static long[] ImprovingQuickSort3(long n, long[] a)
{
    //write your code here
    return new long[] { 0 };
}
```

```
public static string ProcessImprovingQuickSort3(string inStr) =>
    TestTools.Process(inStr, (Func<long, long[], long[]>)ImprovingQuickSort3);
```

```
[TestMethod(), Timeout(1000)]
[DeploymentItem(@"TestData", "A5_TestData")]
0 references | 0 changes | 0 authors, 0 changes
public void Graded_ImprovingQuickSort3Test()
{
    TestTools.RunLocalTest("A5", Program.ProcessImprovingQuickSort3, "TD3");
}
```

۴ Number of Inversions

inversion در یک دنباله از اعداد مانند $a_0, a_1, ..., a_n$ یعنی به ازای $0 <= i < j < n$ رابطه ی $a_i > a_j$ برقرار باشد. شما در این سوال باید تعداد inversion های آرایه ی ورودی را پیداکنید. تعداد Inversion یک آرایه مشخص می کند که چقدر یک آرایه مرتب شده است. یعنی در واقع در یک آرایه ی نزولی تعداد inversion ها صفر می باشد.

testcase های این تمرین هم به این شکل است که در فایل ورودی، خط اول تعدا المان های آرایه و خطوط بعدی المان های آرایه می باشد و در فایل خروجی یک عدد می باشد که برابر با تعداد inversion های آرایه است.

- محدودیت زمانی : **۱۰۰۰ میلی ثانیه**

اسامی توابع شما در این تمرین باید به صورت زیر باشد.

- Algorithm Function : NumberofInversions4
- Process Function : ProcessNumberofInversions4
- Test Function : Graded_NumberofInversions4Test

```
public static long NumberofInversions4(long n, long[] a)
{
    //write your code here
    return 0;
}
```

```
public static string ProcessNumberofInversions4(string inStr) =>
    TestTools.Process(inStr, (Func<long, long[], long>)NumberofInversions4);
```

```
[TestMethod(), Timeout(1000)]
[DeploymentItem(@"TestData", "A5_TestData")]
| 0 references | 0 changes | 0 authors, 0 changes
public void Graded_NumberofInversions4Test()
{
    TestTools.RunLocalTest("A5", Program.ProcessNumberofInversions4, "TD4");
}
```

۵ Organizing a Lottery

فرض کنید سازماندهی یک olnline Lottery به شما واگذار شده است. برای شرکت در مسابقه، هر شرکت کننده یک عدد صحیح را انتخاب می کند. سپس شما به صورت تصادفی چندین بازه را تعریف می کنید. امتیاز هر شرکت کننده در lottery متناسب با تعداد بازه هایی است که شامل عدد مورد نظر شرکت کننده است منهای تعداد محدوده هایی است که آن را شامل نمی شود. شما نیاز به یک الگوریتم کارآمد برای محاسبه امتیاز هر شرکت کننده دارید. Algorithm Naive برای حل این سوال این است که برای همه شرکت کنندگان همه بازه ها را اسکن کنید. اما در این قرعه کشی هزاران نفر از شرکت کنندگان و هزاران محدوده وجود دارد. به همین دلیل شما به یک الگوریتم سریع نیاز دارید.

صورت مسئله به بیان ریاضی به صورت زیر است:

فرض کنید دنباله ای به طول n از اعداد در اختیار دارید که هر کدام از المان های این دنباله یک نقطه هستندو دنباله ی دیگری هم به طول m داریم که شامل بازه ای از اعداد یا segment می باشد. شما باید یک الگوریتم divide and conquer بنویسیدکه برای هر نقطه، تعداد segment هایی که شامل آن نقطه می شود را خروجی بدهد.

فرمت testcase های این سوال به این صورت است که در فایل ورودی، خط اول شامل دنباله نقطه هاست و در هر یک از خطوط بعدی، عدد اول شروع بازه و عدد دوم پایان بازه است. فایل خروجی هم تعداد segment هایی که شامل یک نقطه می شود؛ را نشان می دهد.

- محدودیت زمانی : **۱۰۰۰ میلی ثانیه**

اسامی توابع شما در این تمرین باید به صورت زیر باشد.

- Algorithm Function : OrganizingLottery5
- Process Function : ProcessOrganizingLottery5
- Test Function : Graded_OrganizingLottery5Test

```
public static long[] OrganizingLottery5(long[] points, long[] startSegments,
    long[] endSegment)
{
    //write your code here
    return new long[] { 0 };
}


public static string ProcessOrganizingLottery5(string inStr) =>
    TestTools.Process(inStr,OrganizingLottery5);
```

```
[TestMethod(), Timeout(1000)]
[DeploymentItem(@"TestData", "A5_TestData")]
| 0 references | 0 changes | 0 authors, 0 changes
public void Graded_OrganizingLottery5Test()
{
    TestTools.RunLocalTest("A5", Program.ProcessOrganizingLottery5, "TD5");
}
```

۶ Closest Points

فرض کنید n تا نقطه داریم. شما باید در این مجموعه دو نقطه را پیدا کنید که نزدیک ترین فاصله را از یکدیگر دارند. این مسئله یکی از کاربردی ترین مسائل در حوزه ی گرافیک، بینایی کامپیوتر و کنترل ترافیک است.
فرمت testcase های این سوال به این صورت است که در فایل ورودی در خط اول تعداد نقطه ها و در خطوط بعدی به ترتیب عدد اول x، نقطه و عدد دوم y نقطه است. فایل خروجی هم شامل یک عدد است که همان نزدیک ترین فاصله است.

- محدودیت زمانی : **۱۰۰۰ میلی ثانیه**

اسامی توابع شما در این تمرین باید به صورت زیر باشد.

- Algorithm Function : ClosestPoints6
- Process Function : ProcessClosestPoints6
- Test Function : Graded_ClosestPoints6

```
public static double ClosestPoints6(long n, long[] xPoints, long[] yPoints)
{
    //write your code here

    return 0;
}
```

```
public static string ProcessClosestPoints6(string inStr) =>
   TestTools.Process(inStr, (Func<long, long[], long[], double>)
       ClosestPoints6);
```

```
[TestMethod(), Timeout(1000)]
[DeploymentItem(@"TestData", "A5_TestData")]
| 0 references | 0 changes | 0 authors, 0 changes
public void Graded_ClosestPoints6()
{
    TestTools.RunLocalTest("A5", Program.ProcessClosestPoints6, "TD6");
}
```