

Instructions: This exam is to be taken in silence, without notes, books, or electronic devices (including "smart" watches or earbuds). The time limit to complete it is the duration of the class period (1 hour and 15 minutes). Answer the following questions and problems, trying to be as clear and as accurate as possible. Take your time to read the statements carefully before trying to answer them. If you need more space, write on the back of your test page and indicate it clearly. When writing code, make sure your special punctuation characters are legible, and your lowercase and uppercase letters are easy to distinguish. As usual, every statement or series of statement is assumed to be in a valid class and method, and you can use the C.RL(), C.W() and C.WL() abbreviations. You can also abbreviate exceptions' names: for instance, write A00RE for ArgumentOutOfRangeException.

___ / 10 pts.

Problem 1 Assume arrayEx is an array of int containing the following values:

 | 10 | 15 | 30 | 45 | 60 | 75 | 90 | 105 | 120 | 135 | 150 | 165 | 180 |

and consider the following algorithm:

```
bool fsf = false;
int tar = 105;
int sta = 0;
int end = arrayEx.Length - 1;
int mid, cur;
while (sta <= end && !fsf)
{
    mid = (sta + end) / 2;
    cur = arrayEx[mid];
    if (tar == cur){fsf = true;}
    else if (tar > cur){sta = mid + 1;}
    else {end = mid - 1;}
}
```

Complete the following table, giving the value of sta, end, mid, fsf and cur before the while loop is executed, after it has executed one time, two times, etc. If the value is not set at this point, write "undef", and if the loop stops before the Xth iteration, simply cross out the Xth iteration. Report the value even if it did not change.

| | Before loop | After 1 iteration | After 2 iterations | After 3 iterations | After 4 iterations |
|-----|-------------|-------------------|--------------------|--------------------|--------------------|
| sta | 0 | 7 | 7 | 7 | / |
| end | 12 | 12 | 8 | 8 | |
| mid | undef | 6 | 9 | 7 | |
| fsf | false | false | false | true | |
| cur | undef | 30 | 135 | 105 | |

— / 30 pts.

Problem 2 We implement methods for a simple Calculator **static** class and use one of them. The header of the methods is always of the form **public static int XXX(...)** with XXX the name of the method, and ... a list of argument and their datatype. Give explicitly the headers when providing methods in the following questions.

1. Write a Square method that takes one `int` argument and returns its product if the parameter is less than 46341¹, and throw an `OverflowException` exception otherwise.

```
public static int Square (int p) {
    if (p > 46341) throw new OER();
    return p * p;
}
```

2. Write a Divide method that takes two `int` arguments and returns the result of dividing the first parameter by the second. If the second parameter is 0, then the method should throw an `ArgumentOutOfRangeException` exception. If the result of the division is not "whole", then the method should throw an `ArgumentException` exception. For example, since the result of dividing 10 by 3 is 3.3333..., the method should raise an `ArgumentException` exception instead of returning 3.

```
public static int Divide (int x, int y) {
    if (y == 0) throw new AORE();
    if (x % y != 0) throw new AE();
    return x / y;
}
```

3. Write a Product method that takes two `int` arguments and returns the result of multiplying the first parameter by the second. If the result of the multiplication may exceed `int.MaxValue`, then an `OverflowException` should be thrown².

```
public static int Product (int x, int y) {
    if (x > int.MaxValue / y) throw new OER();
    return x / y;
}
```

¹The product of 46,341 is greater than the maximum value an `int` can hold, `int.MaxValue`.

²Keep in mind that for any two `int` variables `x`, `y`, the condition `x * y > int.MaxValue` will always be **false**. Indeed, if `x * y` exceeds `int.MaxValue`, then the value will simply overflow. You need to be creative!

4. Write an Additive method that takes two int arguments and a string argument:

- If the string is "+", return the sum of the two int arguments.
- If the string is "-", return the difference of the two int arguments.
- If the string is **null**, raise an `ArgumentNullException` exception.
- If the string is anything else, raise an `ArgumentException` exception.

```
public static int Additive(int x, int y, string o) {
    if (o == null) throw new ANE();
    if (o != "+" && o != "-") throw new AE();
    if (o == "+") return x + y;
    return x - y;
}
```

5. Write a piece of code (to be inserted into a `Main` method) that asks the user to enter two numbers and an operation sign, and either call the `Additive` method and display the value it returns if the user indeed entered two integers and a string, or raise an exception if the user did not begin by entering two numbers (no need to loop). Your code should *not* check the value of the string entered for the operation, but must catch the exceptions potentially thrown by the `Additive` method. Your code should also display "Thanks!" regardless of whether an exception was thrown or not.

```
try { C.WL("Enter 2 numbers & an additive operator.");
```

```
    int a1 = int.Parse(C.RL());
```

```
    int a2 = int.Parse(C.RL());
```

```
    string o = C.RL();
```

```
    calculator.Additive(a1, a2, o);
```

```
} catch (ANE) { C.WL("wrong input"); }
```

```
catch (AE) { C.WL("not an additive operator."); }
```

```
catch (FormatException) { C.WL("Not a number"); }
```

```
finally { C.WL("Thanks!"); }
```

*int.Parse may
throw that exception.*

___ / 20 pts. **Problem 3** In this problem, we will implement a SDCard class to represent SD cards. Add attributes to your answer if needed.

1. Implement a Nickname string property using automatic properties.

```
public string NickName { get; set; }
```

2. Implement a Capacity int property whose setter raises an ArgumentException exception if the value passed as argument is not 8, 16, 32, 64 or 128. The getter should simply return the value stored.

```
private int capacity;
public int Capacity { set { if (value == 8 || value == 16 ||
value == 32 || value == 64 || value == 128) capacity = value;
else throw new AE(); }
get { return capacity; } }
```

3. Implement a CapacityInGb int property with only a getter, that returns the Capacity times 8.

```
public int CapacityInGB { get { return capacity * 8; } }
```

4. Implement a ToString method that returns a string containing the nickname of the SD card, its capacity in gigabytes (GB, from question 2.) and gigabits (Gb, from question 3.).

```
public override string ToString () {
return "name: " + Nickname + "\nCapacity" + Capacity
+ "GB\nCapacity in Gb: " + CapacityInGB
+ "Gb." ; }
```

— / 20 pts.

Problem 4 Suppose given a 2-dimensional rectangular array of `int` named `matrix`. Write code that will display "There is one." if there is *exactly one* element stored in `matrix` whose value is the sum of its indices. For example, if `matrix` contains `{{5, 2, 3}, {1, 0, 2}}`, then the message should be displayed as only 1 is the sum of its row and column indices. However, if `matrix` contains `{{0, 3}, {3, 2}}`, then the message should not be displayed, as both 0 and 2 are the sums of their row and column indices.

```
int count = 0;
for (int row = 0; row < matrix.GetLength(0); row++) {
    for (int col = 0; col < matrix.GetLength(1); col++) {
        if (matrix[row, col] == col + row) count++;
    }
}

if (count == 1) { Console.WriteLine("There is one"); }
```

— / 20 pts. **Problem 5** Answer the following short questions:

1. What would be displayed by the following?

```
List<int> vList = new List<int>() { 1, 2, 3 };
vList.Add(10);
vList[vList.Count - 1] = vList.Count;
vList.RemoveAt(0);
vList.Remove(2);
foreach (int elem in vList){Console.WriteLine(elem);}
```

3 ↙
4 ↙

2. Write, using a generic type parameter, a First static method that returns the first element of an array of any datatype.

```
public static T First<T>(T[] aP){
    return aP[0];
}
```

3. Given an array of int arrayP and an int target variable, write the code that performs a linear search: it should display "Found it!" if target is in arrayP.

```
for (int i = 0; i < arrayP.Length; i++) {
    if (arrayP[i] == target) { c.wl ("Found it!"); }
}
```

4. Write the statements to declare a jagged array containing first an array containing 10 int, then an array containing 5 int.

```
int[][] jAr = new int[2][];
jAr[0] = new int[10];
jAr[1] = new int[5];
```

5. In our custom implementation of lists, which of the following correctly remove the first element of a CList?

- public void RemoveF(){first = first.Next;}
- public CList RemoveF(){return first.Next;}
- public void RemoveF(){if (first != null) first = first.Next;}
- public void RemoveF(){if (first != null) first.Data = first.Next.Data;}