Review

R104 – Random Number Generation

Megabytes

What is a megabyte?

Extension: In what situations is it more beneficial to use different definitions?

Megabytes

According to the exam board:

- Megabyte: 1 million bytes (1×10^6)
- Mebibyte: 1024 kibibytes (1×2^{10})
- Kibibyte: 1024 bytes

Starter

```
class Program
{
  static int num;
  static void DoIt()
  {
    Console.WriteLine(num);
  }
  static void Main(string[] args)
  {
    Console.Write("Enter a number: ");
    int num = int.Parse(Console.ReadLine());
    DoIt();
    Console.ReadKey();
  }
}
```

Debugging

Syntax, runtime and logical errors

Syntax errors

```
static void Main(string[] args)
{
    Console.WriteLine("Can you find all my errors?");
    myerror = Tru
    Console.Readk
```

'myerror' is not declared. It may be inaccessible due to its protection level.

Show potential fixes (Alt+Enter or Ctrl+.)

Syntax errors

```
static void Main(string[] args)
{
    Console.WriteLine("Can you find all my errors?");
    myerror = Tru
    Console.Readk
```

'myerror' is not declared. It may be inaccessible due to its protection level.

Show potential fixes (Alt+Enter or Ctrl+.)

- Occur before the program is run
- Code cannot be compiled/built, let alone run
- Indicated by a ‘red squiggly’

Syntax errors

```
static void Main(string[] args)
{
    Console.WriteLine("Can you find all my errors?");
    myerror = Tru
    Console.Readk
```

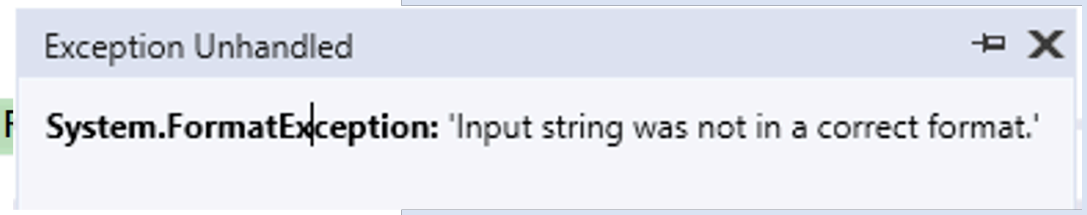
'myerror' is not declared. It may be inaccessible due to its protection level.

Show potential fixes (Alt+Enter or Ctrl+.)

- Occur before the program is run
- Code cannot be compiled/built, let alone run
- Indicated by a ‘red squiggly’
- The code itself is invalid in the language
- Error messages are typically more descriptive
- Fixed by rewriting code to be valid in the syntax of the language

Runtime errors/exceptions

```
static void Main(string[] args)
{
    int number;
    number = int.Parse(Console.R
    Console.ReadKey();
}
```



- Occur while the program is running
- Occur when the program tries to do something it shouldn't
  - e.g. assigning a string to an integer variable
  - e.g. representing a number that is too large
- Typically cause the program to crash

Logic errors

- Occur when the program is running
- Often do not crash the program
- Cause results that are different from what is expected

Debugging Demo

Worksheet

L116 – Debugging Worksheet

This is on Google Classroom.

Copy the code in to a new project and carefully list all of the syntax, runtime and logical errors.

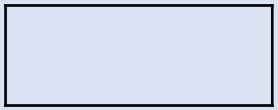
Test the program thoroughly to make sure it works for all inputs.

Arrays in C#

Arrays in C#

Single variable declaration

```
int num;
```



Array declaration

```
int[] num;
num = new int[7];
```

	0
	1
	2
	3
	4
	5
	6

Looping through arrays

```
int[] student = new int[16];

for (int i = 0; i < 16; i++)
{
  student[i] = 1;
}
```

student															
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

Looping through arrays

```
int[] student = new int[16];

for (int i = 0; i < 16; i++)
{
  student[i] = 1;
}
```

student															
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**

Looping through arrays

```
int[] student = new int[16];

for (int i = 0; i < 16; i++)
{
  student[i] = i;
}
```

student															
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

Looping through arrays

```
int[] student = new int[16];

for (int i = 0; i < 16; i++)
{
  student[i] = i;
}
```

student															
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**

Populating an array

Method 1 – One index at a time

```
string[] names = new string[5];
names[0] = "Kai";
names[1] = "Reuben";
names[2] = "Heidi";
names[3] = "Giulia";
names[4] = "Elias";
```

Populating an array

Method 2 – Using a for loop

```
string[] names = new string[5];

for (int i = 0; i < names.Length; i++)
{
  names[i] = Console.ReadLine();
}
```

Populating an array

Method 3 – Auto-initialise

```
string[] names = {"Orlando", "Lev", "Sienna", "Avery"};
```

Populating an array

Method 3 – Auto-initialise

```
string[] names = {"Orlando", "Lev", "Sienna", "Avery"};
```

What’s nice about this method?

Populating an array

Method 3 – Auto-initialise

```
string[] names = {"Orlando", "Lev", "Sienna", "Avery"};
```

What's nice about this method? We don't have to specify the size explicitly

Exercise

You have 10 minutes:

1. Declare an integer array of size 8. Using a **for loop**, put the value 2 into every cell in the array. Display this in the console.
2. Auto-initialise a string array with 5 names. Display the names in the console **in order**.
3. Display the names in the console **in reverse order**.
4. Ask for a character from the user. Count the number of occurrences of that character in each of the names, storing the results in a new integer array.
5. Using this new array, find the mean number of occurrences across all 5 names. Print this mean to the console.

Worksheet

Arrays PRIMM (Google Classroom)

or W108 – C# Arrays if you're feeling confident

Adventure Games

- Start/continue working on your adventure games
- You will be playing and evaluating each others' adventure games in the second lesson after half term
  - Make sure you bring them with you to that lesson