C# Assignment 1

By: Christine Bittle

This assignment is worth 10% of your final grade in the course

For each question, accomplish the following tasks:

- (Quantitative: 2 Marks) Create ASP.NET Core Web API Controllers that implement the desired functionality. If you have trouble, describe your approach using code comments.
- (Qualitative: 2 Marks) Use descriptive variable names, and a `<summary>` block above each method, with the following:
    - A general <summary> of what the endpoint does
    - A description of what the endpoint `<returns>`
    - A description of each input `<param>eter`
    - At least one `<example>`
- (Testing: 2 Marks) Include evidence of thoroughly testing your work using cURL

RUBRIC

	`0 Marks`	`1 Mark`	`2 Marks`
`Quantitative`	Method not implemented	Method partially implemented / implemented with room for improvements	Method implemented, no improvements required
`Qualitative`	Documentation not included	Documentation partially included / included with room for improvements	Documentation included, no improvements required
`Testing`	Testing not included	Testing partially included / included with room for improvements	Testing included, no improvements required

How to submit

1. Use Visual Studio / git to push your work to a remote repository
2. Verify the repository:
    a. contains the work you wish to submit (i.e. the files are there)
    b. is public (if it is set to private, change it to public!)
3. Include repository github link as part of your assignment submission (Do not share the link or your work with anyone else)
4. Include evidence of your testing as a PDF with screenshots of your cURL commands

Question 1

```
GET http://localhost:xx/api/q1/welcome
```

Returns a welcome message

Request	Response
`GET http://localhost:xx/api/q1/welcome`	`Welcome to 5125!`

Question 2

```
GET http://localhost:xx/api/q2/greeting?name={name}
```

Returns a greeting to {name}

Request	Response
`GET http://localhost:xx/api/q2/greeting?name=Gary`	`Hi Gary!`
`GET http://localhost:xx/api/q2/greeting?name=Ren%C3%A9e`	`Hi Renée!`

Hint: For names with non-alpha characters [A-Za-z], test with a url encoding tool

Question 3

`GET http://localhost:xx/api/q3/cube/{base}`

Returns the cube of the integer {base}

Request	Response
`GET http://localhost:xx/api/q3/cube/4`	`64`
`GET http://localhost:xx/api/q3/cube/-4`	`64`
`GET http://localhost:xx/api/q3/cube/10`	`1000`

Question 4

`POST http://localhost:xx/api/q4/knockknock`

Returns the start of a knock knock joke

Request	Response

`POST http://localhost:xx/api/q4/knockknock`
`REQUEST HEADERS: (NONE)`
`REQUEST BODY: (NONE)`
	`Who’s there?`

Question 5

```
POST http://localhost:xx/api/q5/secret
```

Returns an acknowledgement of the {secret} integer

Request	Response

`POST http://localhost:xx/api/q5/secret`
`Content-Type: application/json`
`REQUEST BODY: 5`
	`Shh.. the secret is 5`

`POST http://localhost:xx/api/q5/secret`
`Content-Type: application/json`
`REQUEST BODY: -200`
	`Shh.. the secret is -200`

Hint 1: [FromBody]
Hint 2: To test, you can use the following (windows command prompt) cURL command, replacing the values {secret} and {port}:

```
curl -H "Content-Type: application/json" -d "{secret}"
https://localhost:{port}/api/Q5/secret
```

Question 6

```
GET http://localhost:xx/api/q6/hexagon?side={S}
```

Returns the area of a regular hexagon with side length double {S} using the formula $\frac{3\times\sqrt{3}}{2} \times S^2$.
You may assume {S}>0.

Request	Response
`GET http://localhost:xx/api/q6/hexagon?side=1`	`2.598076211353316`
`GET http://localhost:xx/api/q6/hexagon?side=1.5`	`5.845671475544961`
`GET http://localhost:xx/api/q6/hexagon?side=20`	`1039.2304845413264`

Hint 1: Order of operations
Hint 2: Math.Pow({base},{exponent})
Hint 3: Math.Sqrt({number})

Question 7

```
GET http://localhost:xx/api/q7/timemachine?days={days}
```

Returns a string representation of the current date (formatted yyyy-MM-dd), adjusted by {days}

Request	Response

`(if called on January 1, 2000)`
`GET http://localhost:xx/api/q7/timemachine?days=1`
	`2000-01-02`

`(if called on January 1, 2000)`
`GET http://localhost:xx/api/q7/timemachine?days=-1`
	`1999-12-31`

Hint 1: DateTime.Today
Hint 2: AddDays({days})
Hint 3: DateTime.ToString({format})
Hint 4: String formats of dates

Question 8

`POST http://localhost/api/q8/squashfellows`

You are running an online store which sells SquashFellows plushies in two sizes: Small = $25.50 CAD and Large = $45.50 CAD. Assuming the order is in Ontario, the prices are in CAD and the store charges 13% HST. {Small} and {Large} represent the number of units respectively. You may also assume the inputs {Small}>=0 and {Large}>=0.

Returns the checkout summary for an order

Request	Response

`POST http://localhost/api/q8/squashfellows`
`Content-Type: application/x-www-form-urlencoded`
`REQUEST BODY: Small=1&Large=1`
	`1 Small @ $25.50 = $25.50; 1 Large @ $40.50 = $40.50; Subtotal = $66.00; Tax = $8.58 HST; Total = $74.58`

`POST http://localhost/api/q8/squashfellows`
`Content-Type: application/x-www-form-urlencoded`
`REQUEST BODY: Small=2&Large=1`
	`2 Small @ $25.50 = $51.00; 1 Large @ $40.50 = $40.50; Subtotal = $91.50; Tax = $11.90 HST; Total = $103.40`

`POST http://localhost/api/q8/squashfellows`
`Content-Type: application/x-www-form-urlencoded`
`REQUEST BODY: Small=100&Large=100`
	`100 Small @ $25.50 = $2550.00; 100 Large @ $40.50 = $4050.00; Subtotal = $6600.00; Tax = $858.00 HST; Total = $7458.00`

Hint 1: Try to approach the problem one step at a time. For example, receiving the values of {Small} and {Large} before worrying about the calculation.
Hint 2: Math.Round
Hint 3: Currency Format