Here is a C# Basics Cheat Sheet covering fundamental concepts, syntax, and examples.

Table of Contents

1. Hello World (Basic Syntax)

```
using System;
```

```
class Program {
    static void Main() {
        Console.WriteLine("Hello, World!"); // Output text
    }
}
```

- using System; → Includes standard libraries.
- Main() → Entry point of the program.
- Console.WriteLine() → Prints text to the console.

2. Variables & Data Types

```
int age = 25;       // Integer
double price = 9.99;// Decimal number
char grade = 'A';   // Single character
string name = "John"; // Text
bool isActive = true; // Boolean
```

- int → Whole numbers.
- double → Floating-point numbers.
- char → Single characters.
- string → Text.
- bool → True or False.

---

3. Constants

```
const double PI = 3.14159; // Cannot change after assignment
```

4. Operators

Operator	Description	Example
+	Addition	a + b
–	Subtraction	a – b
*	Multiplication	a * b
/	Division	a / b

Operator	Description	Example
%	Modulus	a % b
++	Increment	a++
—	Decrement	a–

5. Conditionals

```
int num = 10;
if (num > 5) {
    Console.WriteLine("Greater than 5");
} else if (num == 5) {
    Console.WriteLine("Equal to 5");
} else {
    Console.WriteLine("Less than 5");
}
```

6. Switch Case

```
char grade = 'A';

switch (grade) {
    case 'A':
        Console.WriteLine("Excellent!");
        break;
    case 'B':
        Console.WriteLine("Good!");
        break;
    default:
        Console.WriteLine("Invalid grade");
        break;
}
```

7. Loops

For Loop

```
for (int i = 0; i < 5; i++) {
```

```
    Console.WriteLine(i);
}
```

While Loop

```
int count = 0;
while (count < 5) {
    Console.WriteLine(count);
    count++;
}
```

Do-While Loop

```
int num = 0;
do {
    Console.WriteLine(num);
    num++;
} while (num < 5);
```

8. Arrays

```
int[] numbers = { 1, 2, 3, 4, 5 };
Console.WriteLine(numbers[0]); // Outputs 1
```

9. Methods (Functions)

```
static void Greet(string name) {
    Console.WriteLine("Hello, " + name);
}

Greet("Alice");
```

- static → Methods must be static inside a static class.
- Parameters → Passed inside ().
- Return Value Example

```
static int Add(int a, int b) {
    return a + b;
}
```

```
Console.WriteLine(Add(3, 4)); // Outputs 7
```

10. Classes & Objects

```
class Car {
    public string brand;

    public Car(string b) {
        brand = b;
    }
}

Car myCar = new Car("Toyota");
Console.WriteLine(myCar.brand);
```

- Class → Defines a blueprint.
- Object → Instance of a class.

11. Encapsulation (Getters & Setters)

```
class Person {
    private string name;

    public void SetName(string newName) {
        name = newName;
    }

    public string GetName() {
        return name;
    }
}

Person p = new Person();
p.SetName("Alice");
Console.WriteLine(p.GetName()); // Alice
```

12. Inheritance

```
class Animal {
    public void MakeSound() {
        Console.WriteLine("Some sound");
    }
}

class Dog : Animal {
    public void Bark() {
        Console.WriteLine("Woof!");
    }
}

Dog myDog = new Dog();
myDog.MakeSound(); // Inherited method
myDog.Bark();      // Dog's own method
```

13. Polymorphism

```
class Animal {
    public virtual void Speak() {
        Console.WriteLine("Animal sound");
    }
}

class Dog : Animal {
    public override void Speak() {
        Console.WriteLine("Bark!");
    }
}

Animal myAnimal = new Dog();
myAnimal.Speak(); // Outputs "Bark!"
```

14. Interfaces

```
interface IAnimal {
    void MakeSound();
}

class Dog : IAnimal {
    public void MakeSound() {
        Console.WriteLine("Bark!");
    }
}

Dog myDog = new Dog();
myDog.MakeSound();
```

15. Exception Handling

```
try {
    int x = 10 / 0; // Error
} catch (Exception e) {
    Console.WriteLine("Error: " + e.Message);
} finally {
    Console.WriteLine("This runs no matter what.");
}
```

16. Reading User Input

```
Console.Write("Enter your name: ");
string userName = Console.ReadLine();
Console.WriteLine("Hello, " + userName);
```

17. File Handling

Writing to a File

```
using System.IO;
File.WriteAllText("test.txt", "Hello World!");
```

Reading from a File

```
string content = File.ReadAllText("test.txt");
Console.WriteLine(content);
```

18. Lists

```
using System.Collections.Generic;

List<string> names = new List<string>() { "Alice", "Bob" };
names.Add("Charlie");
Console.WriteLine(names[2]); // Outputs Charlie
```

19. Dictionaries

```
using System.Collections.Generic;

Dictionary<string, int> ages = new Dictionary<string, int>();
ages["Alice"] = 25;
ages["Bob"] = 30;

Console.WriteLine(ages["Alice"]); // 25
```

20. Asynchronous Programming

```
using System;
using System.Threading.Tasks;

class Program {
    static async Task Main() {
        await DoWork();
        Console.WriteLine("Main Finished");
    }

    static async Task DoWork() {
        await Task.Delay(2000);
        Console.WriteLine("Work Done");
    }
```

```
}
```

Conclusion

This C# Cheat Sheet covers:

Basic syntax
Variables, loops, conditionals
OOP principles (Encapsulation, Inheritance, Polymorphism)
Exception handling & file handling
Asynchronous programming