C#

```
Public Class Car
{
    public string rightfront;
    public string leftfront;
    public string rightRear;
    public string leftRear;

    public Car (string A, string B, string C, string D)
    {
        rightfront = A;
        leftfront = B;
        rightRear = C;
        leftRear = D;
    }

    public void Pits()
    {
        console.writeline("The car is enterinring to pits and the tyres will be change");
    }
}

Public abstract class Tyres : Car
{
    public int durabilityMax;
    public bool wet;
    public string color;

    public Tyres (string A, string B, string C, string D, int lapsMax, bool isWet, string tyreColor):base
    (A, B, C, D)
    {
        durabilityMax = lapsMax;
        wet = isWet;
        color = tyreColor;
    }

    public virtual string Durability(int laps)
    {
        return "The tyres have been on track " + laps + " laps";
    }

    public void showAll()
    {
        console.writeLine($"The tyres are {rightfront}, {leftfront}, {rightRear}, {leftRear}
    }

    public abstract void PSI();
}
```

```
Public class Soft : Tyres
{
  public soft (string A, string B, string C, string D, int lapsMax, bool isWet, string
  tyreColor) : base (A, B, C, D, lapsMax, isWet, tyreColor)
  {
  {

  public override string Durability (int laps)
  {
    if (laps>durabilityMax)
    {
      Pits();
      return "The tyres have been on track " + laps + "laps \n The Softs are been change";
    }
    else
    {
      return "The tyres have been on track "+laps+"laps \n Can continue";
    }
  }

  public override void PSI() {
    Console.writeline ("The air pressure required is 34");
  }
}

Public class Medium : Tyres
{
  public Medium (string A, string B, string C, string D, int durabilityMax, bool isWet, string
  tyreColor) : base (A,B,C,D, lapsMax, isWet, tyreColor)
  {
  }

  public override string Durability (int laps)
  {
    if (laps > durabilityMax)
    {
      Pits();
      return "The tyres have been on track "+laps+"laps \n The mediums are been change";
    }
    {
      Return "The tyres have been on track "+laps+"laps \n Can continue";
    }
  }

  public override void PSI()
  {
    Console.writeline ("The air pressure required is 40");
  }
```

```
Public class Hard : Tyres
{
  public Hard (string A, string B, string C, string D, int lapsMax, bool isWet, string tyreColor) : base
  (A,B,C,D, lapsMax, isWet, tyreColor)
  {
  }

  public override string Durability (int laps)
  {
    if (laps > durabilityMax)
    {
      Pits();
      return "The tyres have been on track "+laps+"laps \n The hards are been change";
    }
    else
    {
      return "The tyres have been on track"+laps+"laps \n Can continue";
    }
  }

  public override void PSI()
  {
    Console.writeline ("The air pressure required is 61");
  }
}


Public class Intermedium : Tyres
{
  public Intermedium (string A, string B, string C, string D, int lapsMax, bool isWet,
  string tyreColor) : base (A,B,C,D, lapsMax, isWet, tyreColor)
  {
  }

  public override string Durability (int laps)
  {
    if (laps > durabilityMax)
    {
      Pits();
      return "The tyres have been on track " + laps + "laps \n The intermediums are been changes";
    }
    {
      return "The tyre have been on track" + laps + "laps \n Can continue";
    }
  }

  public void PSI()
  {
    Console.writeline("The air pressure required is 37");
  }

  public void AreWet()
  {
    cosole.writeline ("The wether is medium");
  }
}
```

```
Public class FullWet : Tyres
{
  public FullWet(string A, string B, string C, string D, int lapsMax, bool isWet, string
  tyreColor) : base(A, B, C, D, lapsMax, tyreColor)
  {
  {

  public override string Durability(int laps)
  {
    if (laps > durabilityMax)
    {
      Pits()
      return "the tyres have been on track" +laps+ "laps \n the fullwets are been changes";
    }
    else
    { return "the tyres have been on track" +laps+ "laps \n Can continue";
  }
  }

  public override void PSI()
  {
    console.writeline("The air pressure required is 33");
  }

  public void AreWet()
  {
    console.writeline("The weather is heavy");
  }
}
```

Java

```
Public class Car {
  public String rightFront;
  public String leftFront;
  public String rightRear;
  public String leftRear;

  public Car (String A, String B, String C, String D) {

    this.rightFront = A;
    this.leftFront = B;
    this.rightRear = C;
    this.leftRear = D;
  }

  public void Pits() {

    System.out.println("The car is entering to pits and the tyres will be change");
  }
}


Public abstract class Tyres extends Car {

  public int durabilityMax;
  public boolean wet;
  public String color;

  public Tyres (String A, String B, String C, String D, int lapsMax, boolean isWet,
  String tyreColor) {

    super(A, B, C, D);
    this.durabilityMax = lapsMax;
    this.wet = isWet;
    this.color = tyreColor;
  }

  public String Durability(int laps) {
    return "the tyres have been on track " + laps + " laps";

  }

  public void ShowAll() {
    System.out.println("The tyres are: " + rightFront + leftFront + rightRear + leftRear +
    "\n Their durability is " + durabilityMax + laps \n The tyres are wet? " + wet
    + "\n tyres color " + color);
  }

  public abstract void PSI();
```

```
Public class Soft extends Tyres{

  public Soft (String A, String B, String C, String D, int lapsMax, boolean isWet, string tyreColor){
    super (A, B, C, D, lapsMax, isWet, tyreColor);
  }

  @Override

  public String Durability (int laps){

    if (laps > durabilityMax) {
      Pits();
      return "The tyres have been on track " + laps + "laps \n The softs are been change";
    }
    else {
      return "The tyres have been on track " + laps + " \n Can continue" ; }

  public void PSI() {
    System.out.Println ("The air pressure required is 34");
  }
}


Public class Medium extends Tyres {

  public Medium (String A, String B, String C, String D, int lapsMax, boolean isWet, string tyreColor){
    super(A, B, C, D, lapsMax, isWet, tyreColor);
  }

  @Override
  public String Durability (int lap){
    if ( laps > durabilityMax){
      Pits()
      return "The tyres have been on track" + laps + "laps \n The mediums are been change" ;}
    else {
      return "The tyres have been on track" + laps + "laps \n Can continue" ;
    }

  public void PSI() {
    System. out. println ("The air pressore required is 40");
  }
```

```
Public class Hard extends Tyres {

    public Hard(String A, String B, String C, String D, int lapsMax, boolean isWet, string tyreColor){
      super(A, B, C, D, lapsMax, isWet, tyreColor);
    }
    @Override
    Public String Durability(int laps) {
      if (laps > durabilityMax){
        Pits()
        return "The tyres have been on track" + laps + "laps \n the Hards are been change";}
      else {
        return "The tyres have been on track" + laps + "laps \n Can continue";}
      }
    @Override
    public void PSI(){
      System.out.println("The air pressure required is 51");}
}

Public class intermedium extends Tyres {

    public Intermedium(String A, String B, String C, String D, int lapsMax, boolean isWet, string tyreColor){
      super(A, B, C, D, lapsMax, isWet, tyreColor);}

    @Override
    public String Durability(int laps){
      if (laps > durabilityMax){
        Pits()
        return "The tyres have been on track" + laps + "laps \n The intermedium are been changes";}
        else{
        return "The tyres have been on track" + laps + "laps \n Can continue";

    @Override
    public void PSI(){
      System.out.println("The air pressure required is 37");}

    public void AreWet(){

      System.out.println("The weather is medium");
    }
}
```

```
public class fullwet extends Tyres {

    public fullwet(String A, String B, String C, String D, int lapsMax, boolean isWet, String tyreColor){
        super(A, B, C, D, lapsMax, isWet, tyreColo);
    }

    @Override
    public String Durability (int laps){
        if(laps > durabilityMax){
            Pits()
            return("The tyres have been on track" +laps+" laps\n The fullwets are been change");}
        else{
            return "The tyres have been on track+laps+"laps \n Can continue" ;}
    }

    @Override
    public void PSI(){
        System.out.println("The air required is 33");
    }

    public void ArcWet(){
        System.out.println("The weather is heavy");
    }
}
```

javascript

```
export default class Car {

  constructor (A, B, C, D) {
     this.rightFront = A;
     this.leftFront = B;
     this.RightRear = C;
     this.leftRear = D;
  }

  Pits () {
     console.log ("The car is entering to pits and the tyres will be changed ");
  }
}

import Car from "./Car.js";

export default class Tyres extends Car {
  constructor (A, B, C, D, lapsMax, isWet, tyreColor) {
     super(A, B, C, D);
     this.durabilityMax = lapsMax;
     this.wet = isWet;
     this.color = tyreColor;
  }

  Durability (laps) {
     return `the tyres have been on track ${laps} laps`;
  }

  ShowAll () {
     console.log(`The tyres are ${this.rightFront}, ${this.leftFront}, ${this.rightRear},
     ${this.leftRear} \n Their durability is ${durabilityMax} laps \n The tyres are
     wet? ${this.wet} \n tyres color ${this.color}`);
  }

  PSI () {
     console.log ("NON");
  }
}
```

```
Impor Tyres from "./tyres.js";

export default class Soft extends Tyres {

  constructor(A, B, C, D, lapsMax, isWet, tyreColor) {
    super(A, B, C, D, lapsMax, isWet, tyreColor);
  }

  Durability (laps) {
    if (laps > this.durabilityMax){
      this.Pits();
      return ("The tyres have been on track ${laps} laps \n The Softs are being
      changed");
    }
    else { return ("The tyres have been on track ${laps} laps \n Can continue");
    }
  }

  PSI () {
    console.log ("The air pressure required is 34");
  }
}

Import Tyres from "./Tyres.js";

export default class medium extends Tyres {
  Constructor (A, B, C, D, lapsMax, isWet, tyreColor){
    super (A, B, C, D, lapsMax, isWet, tyreColor);
  }

  Durability (laps) {
    if (laps > this.durabilityMax){
      this.Pits();
      return(" The tyres have been on track ${laps} laps \n The mediums are being changed");
      };}
    else{
      return ("The tyres have been on track ${laps} laps \n Can continue");
      }
  }

  PSI () {
    console.log ("The air pressure required is 40");
  }
}
```

```
Import Tyres From "./Tyres.js";

Expor default class Hard extend Tyres {
  constructor (A, B, C, D, lapsMax, isWet, tyreColor) {
    super (A, B, C, D, lapsMax, isWet, tyreColor);
  }

  Durability (laps) {
    IF (laps > this.durabilityMax) {
      this.PitS();
      return "The tyres have been on track ${laps} laps \n The hards are being changed";}
    else {
      return "The tyres have been on track ${laps} laps \n Can continue"; }
  }

  PSI () {
    console.log ("The air pressure required is 69");
  }
}


Import Tyres from "Tyres.js";

Export default class Intermedium extends Tyres {
  constructor (A, B, C, D, lapsMax, isWet, tyreColor) {
    super (A, B, C, D, lapsMax, isWet, tyreColor);
  }

  Durability (laps) {
    IF (laps > this.durabilityMax) {
      this. Pits ();
      return "The tyres have been on track ${laps} laps \n The intermedium are being changed"; }
    else {
      return "The tyres have been on track ${laps} laps \n Can continue"; }
  }

  PSI() {
    console.log("The air pressure required is 37");
  }

  AreWet() {
    console.log("The weather is medium");
  }
}
```

```
Import Tyres from "Tyres.js";

Export default class FullWet extends Tyres {
  constructor (A, B, C, D, lapsMax, isWet, tyreColor){
    Super (A, B, C, D, lapsMax, isWet, tyreColor);
  }

  Durability (laps) {
    if (laps > this.durabilityMax){
      this.Psi();
      return "The tyres have been on track ${laps} laps \n The FullWets are being changed"; }
    else {
      return "The tyres have been on track ${laps} laps \n Can continue";
    }
  }

  PSI(){
    console.log ("The air pressure required is 33");
  }

  AreWet() {
    console.log ("The weather is heavy");
  }
```

Go

```
package main
import "fmt"

type Car struct {
    rightfront String
    leftfront string
    rightRear string
    leftRear string
}

func (c *Car) NewCar (A string, B string, C string, D string){
    c.rightfront = A
    c.leftFront = B
    c.rightRear = C
    c.leftRear = D
}

func (c Car) Pits() {
    fmt.Println("The car is entering to pits and the tyres will be change")
}
```

```
Package main
Import "Fmt"

Type Tyres struct {
    Car
    durabilityMax int
    wet bool
    color string
}

func NewTyres(A,B,C,D string, lapsMax int, isWet bool, tyreColor string){
    Car := NewCar{
        rightfront: A
        leftfront: B
        rightRear: C
        leftRear: D
    }
    return Tyres{
        car: Car,
        durabilityMax: lapsMax,
        wet : isWet,
        color: tyreColor,
    }
}

func (t Tyres) Durability(laps int) string{
    return.fmt.Sprintf("The tyres have been on track %d laps", laps)
}

func (t Tyres) ShowAll(){
    fmt.Printf("The tyres are %s %s %s %s \n Their durability is %d laps \n The tyres are wet?
    %t \n tyre color %s \n", t.rightfront, t.leftfront, t.rightRear, t.leftRear, t.durabilityMax, t.wet, t.color)
}

func (t Tyres) PSI(){}
```

```
Package main
Import "fmt"

type Soft struct {
   Tyres
}

func NewSoft (A, B, C, D string, lapsMax int, isWet bool, tyreColor string) Soft {
   tyres := NewTyres (A, B, C, D, lapsMax, isWet, tyreColor)
   return Soft {Tyres: tyres}
}
(
func (s Soft) Durability (laps int) String {
   if laps > s.durabilityMax {
      s.Pits()
      return fmt.Sprintf("The tyres have been on track %d laps \n The softs are being changed", laps)
   }
   return fmt.Sprintf("The tyres have been on track %d laps \n Can continue, laps
}

func (s Soft) PSI() {
   fmt. println ("The air pressure required is 34");
}


Package main
Import "fmt"

Type Medium struct{
   Tyres
}

func NewMedium (A, B, C, D string, lapsMax int, isWet bool, tyreColor string) Soft {
   tyres := NewTyres (A, B, C, D, lapsMax, isWet, tyreColor)
   return medium {Tyres: tyres}
}

func (m Medium) Durability (laps int) string {
   if laps = m.durabilityMax {
      m.Pits()
      return fmt.Sprintf("The tyres have been on track %d laps \n The medium change", laps)
   }
   return fmt.Sprintf("The tyres have been on track %d laps \n Can continue", laps)
}

func (m Medium) PSI() {
   fmt. println ("The air pressure required is 40")
}
```

```
package main
import "fmt"

type Hard struct {
  Tyres
}

func NewHard(A,B,C,D string, lapsMax int, isWet bool, tyreColor string) Hard {
  tyres := NewTyres(A,B,C,D, lapsMax, isWet, tyreColor)
  return Hard {Tyres: tyres}
}

func (h Hard) Durability(laps int) string {
  if laps > h.durabilityMax {
    h.Pits()
    return fmt.Sprintf("The tyres have been on track %d laps \n The Hards are being changed")
    }
  return fmt.Sprintf("The tyres have been on track %d laps \n Can continue", laps)
}

func (h Hard) PSI(){
  fmt.Println("The air pressure required is 31")
}


package main
import "fmt"

Type Intermedium Struct{
  Tyres
}

func NewIntermedium(A,B,C,D string, lapsMax int, isWet bool, tyreColor string) Intermedium {
  tyres := NewTyres(A,B,C,D, lapsMax, isWet, tyreColor)
  return Intermedium {Tyres: tyres}
}

func (i Intermedium) Durability(laps int) string {
  if (laps > i.durabilityMax){
    i.Pits()
    return fmt.Sprintf("The tyres have been on track %d laps \n The Intermediums are being changed", laps)
  }
  return fmt.Sprintf("The tyres have been on track %d laps \n Can continue", laps)
}

func (i Intermedium) PSI() {
  fmt.Println("The air pressure required is 37")
}

func (i Intermedium) AreWet() {
  fmt.Println("The weather is medium")
}
```

```
package main
import "fmt"

type FullWet struct{
    Tyres
}

func NewFullWet(A,B,C,D string, lapsMax int, isWet bool, tyreColor string) FullWet{
    tyres := NewTyres(A, B, C, D, lapsMax, isWet, tyreColor)
    return FullWet{Tyres: tyres}
}

func (f FullWet) Durability(laps int) string{
    if (laps > FullWet.DurabilityMax){
        f.PSI()
        return fmt.Sprintf("The tyres have been on track %d laps \n The full wets are being changed", laps)
    }
    return fmt.Sprintf("The tyres have been on track %d laps \n Can continue", laps)
}

func (f FullWet) PSI(){
    fmt.Println("The air pressure required is 33")
}

func (f FullWet) AreWet(){
    fmt.Println("The weather is heavy")
}
```