

# CP2019s Project 1 : Car Simulator ver. 2050

## I. Overview


The first project on Computer programming is car simulation on various situations. There are three different types of car modes and environment: car driving on a road, airplane flying in a sky, and submarine under the ocean. Your job is to create three different car mode and make those cars to drive on ten different test cases which will be provided. Depending on the car modes, there are different factors to consider while driving, so be aware of constraints and restrictions.

## II. Contents

### A. Vehicle mode & factors

There are three modes in car simulator. Each mode has attributes that represent current status of vehicle. It is advised to make parent class as 'Vehicle' and implements other classes as child of parent class. (NOT MANDATORY)

#### [ Car ]

- Speed
- Energy
- Temperature 
- Humidity
- Solar panel recharge

Car mode has 4 factors (speed, energy, temperature, humidity) and public function (solar panel recharge). All of the attributes are defined private, so you have to make get/set functions to deal with the situation. When in car mode, the solar charging system is activated to charge energy.

#### [Airplane]

- Speed
- Energy
- Temperature
- Humidity
- Altitude

- Oxygen rate
- Air density

Airplane mode has 7 factors (speed, energy, temperature, humidity, altitude, oxygen rate, air density). All of the attributes are defined private, so you have to make get/set functions to deal with the situation.

[Submarine]

- Speed
- Energy
- Temperature
- Depth
- Light(Always on)
- Oxygen rate
- Water flow

Submarine mod has 6 factors (speed, energy, temperature, depth, oxygen rate, water flow) and public function(light). All of the factors are defined private, so you have to make get/set functions to deal with the situation.

## B. Environments (Road / Sky / Ocean)

**In Road environment**, there are 2 external factors (temperature, humidity).

When vehicle faces the road, 2 factors set by initial values of test cases.

Only car mode go through the road environment.

**In Sky environment**, there are 3 external factors (temperature, humidity, air density).

When vehicle faces the sky, 3 factors set by initial values of test cases.

Only airplane mode go through the sky environment.

**In ocean environment**, there are 2 external factors (temperature, water flow).

When vehicle faces the ocean, 2 factors set by initial values of test cases.

Only submarine mode go through the ocean environment.

## C. Initial Values and the Relationships between variables

- Vehicle initial energy: 1000 / initial oxygen level: 100
- Car default speed: 80km / hr
- Airplane default speed: 900km / hr

- Submarine default speed: 20km / hr

Temperature affects energy. (Celcius)

when  $0 < \text{temperature} < 40$ , energy loss 5

temperature  $\geq 40$ , energy loss 10

temperature = 0, energy loss 8

Humidity affects energy.

when humidity  $< 50$ , energy loss 5

humidity  $\geq 50$ , energy loss 8

**(so, in case of car mode, with temp 20 C and humidity 50 -> 13 energy loss every 50km)**

> "Unit" is defined as following: **50km for road, 10km for ocean, 1000km for sky.** (See Implementation Section III)

Humidity affects solar panel recharge.

The charging system only operates with a humidity  $< 50$

Solar panel recharge affects energy.

after recharging, energy get 200 ( Energy cannot be over 1000) (! Check for constraint)

Air density affects speed.

when air density  $\geq 30$ , speed loss 200

air density  $\geq 50$ , speed loss 300

air density  $\geq 70$ , speed loss 500

When vehicle meet the road, oxygen rate set to 100

Altitude affects oxygen rate.

oxygen rate loss 10 per altitude 1000

Depth affects oxygen rate.

oxygen rate loss 5 per depth 50

Light affects energy.

when in submarine mode, the light is always on, it reduces energy 30 **EVERY 10km movement!**

Water flow affects speed.

when water flow  $\geq 30$ , speed loss 3  
water flow  $\geq 50$ , speed loss 5  
water flow  $\geq 70$ , speed loss 10

### **! IMPORTANT**

All loss should be "ADDED UP"

In case of speed loss, oxygen loss, energy loss, etc...

## D. Test Cases

Test case form: R means road, O means ocean, S means sky.

Each test case consists of duration of course, initial values, and if there are any, unexpected situations.

Example: [R50T20H30],[S3000T10H5A2000D20],[X],[O80T0D100W100],[R150T30H50]

The meaning of this test case: The course starts with 50km of road with temperature value 20 and humidity value 30. After 50km, you will face sky with 3000km. Within the sky, the temperature value is 10, humidity(H) is 5, altitude(A) is 2000, air density(D) is 20. After that, you meet unexpected situation(X). You can deal with this unexpected situation or just ignore it. (your choice) After unexpected situation, you will face 80km of ocean.. and so on.

[R50T20H30] : Road (50km) / Temperature (20 C) / Humidity (30)

[S3000T10H5A2000D20] : Sky (3000km) / Temp (10 C) / Humidity (5) / Altitude (2000) / air Density (30)

[X] & [Y] : unexpected situations (if you are not going for the extra credit, **just ignore these signs**)

[O80T0D100W100] : Ocean (80km) / Temp (0 C) / Depth (100m) / Water flow rate (100)

## E. File I/O

Text files containing each test case will be provided. (ex. TC1.txt, TC2.txt ...)

You have to read each files using I/O function(ifstream), and process the input string to properly function.

Between the environment, there are useful indicators.(ex. ‘,’ , ‘[’, ‘]’, ‘R’, ‘T’, ‘H’ ...)

Using test case string, you can make full journey that vehicle can go through.

Read all ten TC files in the start of the program and use them whenever a test case is called.

### III. Implementation

- From the start to the end, your vehicle will be driving a course (from a test case file). The vehicle should change as it faces different situations.
- The vehicle starts on road and will continue the course until three different endings: when the vehicle successfully arrives at the goal line, when the energy is totally gone, when the oxygen level is zero.
- **Either energy or the oxygen level goes to zero will make the vehicle stop and the journey will be forced to be finished.**
- There are ten test cases (ten different courses to choose) and you have to ask for a test case to run.
  - (Screenshot in Section VI.)
- After choosing the course, ask for two different options to choose: drive until next “unit” or “status change”.
  - (Screenshot in Section VI.)
- “Unit” is defined as following: **50km for road, 10km for ocean, 1000km for sky.**
- “Status change” is defined as following: **the arrival, car mode change(from car to airplane etc), empty energy, zero oxygen level, car stop(\* - for extra credit).**
- **First move mode: Drive for next unit**
  - You will drive the vehicle for a single “unit”.
  - Print out the vehicle status after driving a “unit”.
  - (Screenshot in Section VI.)
  - If energy or oxygen level goes below zero, stop the vehicle and print the status.
  - (If there are too little energy or oxygen to go to the next “unit” (but not zero), the vehicle can make to the next “unit” and the remaining energy or oxygen will be gone.)
- **Second move mode: Drive until the next status change**
  - You will drive the vehicle as long as it has no “status change”

- If there are enough resources(energy/oxygen), you can go as far as the environment does not change.
  - (Suppose you have enough energy) There is a road with 500km. With the first mode, you should print 10 times to drive all 500km. But in this second mode, you can drive the whole 500km without printing the status in between.
  - Obviously, the car should stop if resources gone empty, so you should stop the car right after the empty resource status.
- After each move, you should print the current status.
    - (Screenshot in Section VI.)
  - The current status should be printed DIFFERENTLY with the vehicle mode.
    - Car does not has oxygen level status.
    - Airplane does not has depth status.
    - Submarine only has depth status. Etc....
    - The point is that the output should show correct and only related current status (Screenshot in Section VI. look for the difference between car/airplane/submarine)
- Journey will end in three different cases: the arrival, oxygen failure, energy failure.
  - Print the final status on Distance / Energy / Oxygen values (Exact format is on the screenshot in Section VI.)
  - After the whole journey, you should print “Blackbox” which records all data of [“car mode”, “energy”, “oxygen”, “speed”]
    - For car mode, record oxygen level as 100 (Always)
    - This blackbox records only when status changes. (from car to airplane etc...)

## IV. Constraints

### Code constraints

- The journey starts with the “car mode”. (TC also always starts with Road.)
- On car mode, oxygen level is **ALWAYS 100**.
- From airplane & submarine to car mode, oxygen level goes up to 100.
- From airplane to submarine, oxygen level continues (NOT go up to 100). \*vise versa
- **Speed does NOT affect the move “unit”. (50km per move regardless of car speed)**
- No change of temperature / humidity / altitude / air density / depth / water flow as long as the environment does not change. (It’s possible to have a different value at the next environment change)
- Solar panel recharge function ONLY activates after the vehicle is changed to car mode.

- (Ex. one activation right after the vehicle changes from Airplane/Submarine to Car)
- (Ex2. Airplane -> Car (function activates) -> Submarine -> Car(function again activates) -> ... -> )

Program constraints

- Compile with **C++11**
- **All inputs (menu selections, mode selections) are all valid. NO Exception handling required.**

## V. For Extra credit

### A. Unexpected factor will occur in between the courses (in test case)

At the first of the program, you should ask for the handle mode whether you will consider unexpected situations or not.

```
Mode Select(1 for EXTRA, 2 for NORMAL) :
```

There are two different unexpected events: X, Y.

**X** : For road, you will face reindeer, eagle for sky, and shark for ocean. This event occur with [X] mark on test cases. This event has “20%” chance of complete stop of the vehicle. If the vehicle survives the event, 100 energy is decreased (and will continue).

**Y** : The vehicle will get damaged without a reason. This event occur with [Y] mark on test cases. This event has “35%” chance of complete stop of the vehicle. If the vehicle survives the event, following options will occur on 50% chance. For car on the road, solar panel will be destroyed. For airplane on sky and submarine under the ocean, 30 oxygen level will be decreased.

### B. Text graphic of the progress



- Print the graphic after each session report. (After each move mode)
- For the vehicle, use “@”.
- For road, use “=”. Each character represents 50km.

- For ocean, use “~”. Each character represents 10km.
- For sky, use “^”. Each character represents 1000km.

## VI. Result examples

Initial State

```
PJ1.홍길동.2018-00000
Choose the number of the test case (1~10) :
```

Choosing the test case (print the initial status)

TC5 : [R500T20H20],[S3000T10H5A2000D30],[O80T0D100W100] (**not same with actual TC5**)

```
Choose the number of the test case (1~10) : 5
Test case #5.

Current Status: Car
Distance: 0 km
Speed: 0 km/hr
Energy: 1000
Temperature: 20 C
Humidity: 20
Next Move? (1,2)
CP-2018-00000>
```

Select the first move mode :

```
CP-2018-00000>1
Successfully moved to next 50 km
Current Status: Car
Distance: 50 km
Speed: 80 km/hr
Energy: 990
Temperature: 20 C
Humidity: 20
Next Move? (1,2)
CP-2018-00000>
```

Select the second move mode : (Drive all the way until the next status change)



```
CP-2018-00000>2
Successfully moved to next 450 km
Current Status: Car
Distance: 500 km
Speed: 80 km/hr
Energy: 900
Temperature: 20 C
Humidity: 20
Next Move? (1,2)
CP-2018-00000>
```

Select the first move mode: (Car -> Airplane) (Notice that there are **more features** to show)

```
CP-2018-00000>1
Successfully moved to next 1000 km
Current Status: Airplane
Distance: 1500 km
Speed: 700 km/hr
Energy: 890
Oxygen Level: 80
Temperature: 10 C
Humidity: 5
Altitude: 2000 m
Air Density: 30
Next Move? (1,2)
CP-2018-00000>
```

Select the second move mode:

```
CP-2018-00000>2
Successfully moved to next 2000 km
Current Status: Airplane
Distance: 3500 km
Speed: 700 km/hr
Energy: 870
Oxygen Level: 40
Temperature: 10 C
Humidity: 5
Altitude: 2000 m
Air Density: 30
Next Move? (1,2)
CP-2018-00000>
```

Select the first move mode: (Airplane -> Submarine)

```
CP-2018-00000>1
Successfully moved to next 10 km
Current Status: Submarine
Distance: 3510 km
Speed: 10 km/hr
Energy: 832
Oxygen Level: 30
Temperature: 0 C
Depth: 100 m
Water Flow: 100
Next Move? (1,2)
CP-2018-00000>
```

Select the second move mode: (not enough oxygen → DIE)

```
CP-2018-00000>2
Successfully moved to next 30 km
Final Status:
Distance: 3540 km
Energy: 718
Oxygen Level: 0

!FINISHED : Oxygen failure
Blackbox:
Mode: Car > Airplane > Submarine
Energy Level: 900 > 870 > 718
Oxygen Level: 100 > 40 > 0
Speed: 80 > 700 > 10
-----
```

!FINISHED : Oxygen failure    //(when no oxygen)  
!FINISHED : Energy failure    //(when no energy)  
!FINISHED : Arrived            //(when finished the full course)  
!FINISHED : Vehicle stop       // \* for extra credit

## VII. Report

1. Development/Implementation Environment
2. Specific code description (detailed enough to know about the functions and mechanism of whole process)
3. Troubleshooting points while implementing your code (trial and error, things to be careful of, etc)
4. Screenshots of the interactive console (From the start to the end of whole course, add different screenshots in between status changes and various events)
5. **There is NO report length limitation.**

## VIII. Due date

- **Due on May 10th 23:59.**
- **Submit the whole file zipped into one single zip file(your whole source code or sln and the report) to ETL**
- **CAUTION : 20% deduction per day after the submit due date.**
- **NO submit possible after two days (Must submit before May 13th 00:00)**
- **Compile error : ZERO POINT**
- Code file type : \*.h, \*.cpp, \*.sln are allowed.
- Report file type : docx, hwp, pdf are accepted.

## IX. Evaluation

Classification	Criteria	Points
General	Program Execution I/O Format	15
	Code clarity	15
Test case	Total 10 test cases (each - partial point exist)	50 (5)
Report	Clarity	5
	Code / Screenshot Description	15
Error	Compile Error	-100
<b>Total</b>		<b>100</b>
<b>Extra Credit</b>	Unexpected Situation handling	20
	Graphic progress	10

## X. FAQ

**All questions from email will be answered here. Since this document is on-line editable, please check this FAQ before mailing.**

(will be added after e-mail questions...)