TASK

Capstone Project I — Variables and Control Structures

Visit our website

Introduction

WELCOME TO THE FIRST CAPSTONE PROJECT!

This Capstone Project is a milestone in your learning so far! In this task, you will be consolidating the knowledge you have gained and apply it to a real-world situation! This Capstone Project will allow you to demonstrate your competence in using variables, various data types and if/else-if/else statements. It is worth spending time and effort to make this a project that you can be proud of - it could well be the first project that you add to your developer portfolio!

Get in touch

Connect for support

Remember that with our courses, you're not alone! You can contact an expert code reviewer to get support on any aspect of your course.

The best way to get help is to login to Discord at **https://discord.com/invite/hyperdev** where our specialist team is ready to support you.

Our expert code reviewers are happy to offer you support that is tailored to your individual career or education needs. Do not hesitate to ask a question or for additional support!

CAPSTONE PROJECTS

This is your first Capstone Project with Hyperion! Capstone Projects allow you to test your programming skills while creating a developer portfolio. It is essential to understand a programming language or technology for development. However, it is even more important to be able to apply your knowledge to create software that meets unique client specifications. Creating software allows you to highlight your development skills to a prospective employer!

As a developer, you will need to demonstrate your ability to create software that is needed or wanted. You should also be able to improve existing software. Remember, any great design must be functional and meet the specifications provided by the user. A software solution that looks good and works, but doesn't do what the user wants it to, is like creating a bike with square wheels. A bike of this nature would only work in the scenario pictured below, showing a very specific design to demonstrate a physical principle in a science-teaching context, but it wouldn't be much good to get you from A to B. As a programmer, you need to aim to create resilient code that can function across a variety of contexts and that anticipates and plans for potential edge cases and unforeseen circumstances. Resilience testing is an important part of coding.

DEVELOPER PORTFOLIO

Developers who have the edge are those who find ways to apply their newfound skills from the get-go. A **developer portfolio** (a collection of software that you have made) allows you to demonstrate your skills rather than just telling people about them. It's a way of bringing your CV to life and introducing yourself to the world. As you learn more skills and put these into practice, each project that you complete will become more efficient and eye-catching.

This application series offers you the means to create projects for your very own developer portfolio, allowing you to walk away from this course not only with a certificate but, more importantly, with a headstart into your career!

THE TASK AT HAND

As a professional programmer, you will often be faced with tasks and projects where you don't initially know everything you need to know to create a solution. Part of the solutioning process is to get a project brief and then research the context so as to develop the insight necessary to write great code.

In this Capstone Project, you will be creating a Celsius/Fahrenheit/Kelvin converter so that you will be able to convert the temperatures in all directions. In a work context you might need to do some research to inform your work on a project such as this. Here we have helped you by providing the information you need to learn a little bit about Celsius, Fahrenheit and Kelvin, prior to beginning to code a solution.

Celsius

Celsius (or centigrade) is the measurement of temperature where the freezing point of water is 0°C, the boiling point of water at sea level is 100°C and absolute zero is -273°C. However, when Anders Celsius first created the temperature scale in 1742, he intended for it to be the other way around — he wanted the boiling point to be 0°C! It was actually Jean-Pierre Christin who developed the scale as we know it today, which he measured with the mercury thermometer he designed. Numerous other people are suspected to have created a similar temperature scaling system around the same time, including Carl Linnaeus — a Swedish botanist; Pehr Elvius — secretary of the Royal Swedish Academy of Sciences; Daniel Ekström — an instrument maker; and Mårten Strömer — a student of Anders Celsius.

Fahrenheit

Fahrenheit is the measurement of temperature where 32°F is the freezing point of water. 212°F is the boiling point at sea level and absolute zero is -460.°F. 0°F was defined as the freezing point of a solution of ice, water and ammonium chloride. It is the measurement of temperature most commonly used in the United States of America, but most other countries in the world use Celsius. Daniel Gabriel Fahrenheit created this scale in 1724 drawing from original work done by Herman Boerhaave. He based his scale on the intervals of the ammonium chloride solution (0°F), the freezing point of water (32°F), the body's average core temperature (96°F), and water's boiling point at sea level (212°F). Interestingly enough, with our more accurate thermometers today, we now know that the freezing point of the ammonium chloride solution is 4°F, not 0°F!

Kelvin

Kelvin is the measurement of temperature most commonly used in scientific settings. On the Kelvin scale, the freezing point of water is 273K, the boiling point of water at sea level is 373K and absolute zero is 0K. Kelvin uses the same scale as Celsius, it only has 273 added onto it so that absolute zero is, in fact, zero. This metric was established by William Thomsom, first Baron Kelvin, in 1848.

A note from our coding mentor

Dayle

Did you know that the programs that NASA used in the Apollo mission to the moon were less powerful than a pocket calculator? Yes, you read that right!

These ingenious computer systems were able to guide astronauts across 356 000 km of space from the Earth to the moon and return them safely. Today, a USB memory stick is more powerful than the computers that put a man on the moon.

The Apollo Space Shuttle

Instructions

Feel free to refer back to previous material at any point if you get stuck. Remember that if you require further assistance, our expert code reviewers are always more than willing to help you!

A key goal in this project is not only to get your code working according to the specifications provided but also to ensure that your code is readable. Readable code is easy to read and understand. Readable code is easier to maintain and troubleshoot. To make sure that your code is readable, do the following:

1. **Add comments:** A comment is for information that you add to your code file that isn't read by the computer. A comment is not an instruction to the computer, but rather information that clarifies code for human readers. Although this may seem unnecessary at this point in your learning journey, in the workplace programmers tend to be part of a team and so the readability of their code has a direct impact on their ability to work on coding projects with their teammates, and vice versa. Comments describe what the program does and why things are done as they are; employers need to know that you comment your code appropriately if they are to have confidence in your working with other programmers in the organisation.

Consider the code below for an illustration of how this works. This initial example does not include any comments and using a looping structure you have not been introduced to yet, the *for* loop.

```
let rangeNum = Number(prompt("Enter the max number you'd like to go up to: "));
for (i=0;i<=rangeNum;i++){
  if (i % 2 == 0) {
    console.log(i);
  }
}
```

Now, look at the same code with added comments. Note that single-line comments in JavaScript start with double forward-slash (**//**), but multi-line comments start with **/*** and end with ***/**.

```
// This is a comment in an example JavaScript program.
// In this program, the user inputs a number.
/* The program then outputs all the EVEN numbers from 0 to that number, using a for
loop and and if statement.*/

let rangeNum = Number(prompt("Enter the max number you'd like to go up to: "));
for (i=0; i<=rangeNum; i++){
  if (i % 2 == 0) {
    console.log(i);
  }
}
```

See how comments can help? With comments, you don't have to spend much time trying to figure out what the code does. The comments tell you.

Now consider the next example. As you can see, too many comments can actually detract from the readability of your code instead of enhancing it.

```
// This is an example JavaScript program.
// The user inputs a number.
// The program then outputs all the EVEN numbers from 0 to that number, using a for
loop and if statement.

let rangeNum = Number(prompt("Enter the max number you'd like to go up to: "));
// User enters a number
for (i=0;i<=rangeNum;i++){    //  Create a loop that will repeat from i=0 to i =
rangeNum
  if (i % 2 == 0) {          // If the current value of i is even, the if statement
will be True and 'i' will print
    console.log(i);
  }
```

```
}
```

Best practice is to use comments in moderation, adding enough comments in appropriate places to assist another programmer using your code to see what is going on in your program quickly. Avoiding commenting on every single simple thing, and avoid having so many comments you make your fellow programmer's job harder. The more lines of code you eventually have for a program, the more clear the need for comments will become.

Be sure to add appropriate comments to all the code you write from now on!

2. **Use descriptive variable names:** Using meaningful names for variables also improves the readability of your code. If needed, go back and look at the Variables task to review guidelines for creating good variable names.

3. **Use whitespace and indentation** to enhance readability. Programs that have empty lines between units of work are easier to read.

4. Make sure that all output that your program provides to the user is easy to read and understand. Labelling all data that you output is essential to make the data your program produces more user-friendly. For example, compare the readability of the outputs in the images below. Notice how using spacing and labelling the output makes the second output much more user-friendly than the first.: Some useful escape sequences are listed below:
    - \n - Newline
    - \t - Tab

    Output 1:

```
Assign initial tasks, admin, 10 Oct 2019, 25 Oct 2019, No, Use
taskManager.js to assign each team member with appropriate tasks
```

    Versus Output 2:

```
_____________________________________
```

```
Task:                   Assign initial tasks
Assigned to:            admin
Date assigned:          10 Oct 2019
Due date:               25 Oct 2019
Task complete?          No
task description:       Use taskManager.js to assign each team member
with appropriate tasks

_____________________________________
```

Compulsory Task 1

For this task, you will be creating a temperature metric converter.

- Create a new HTML file in this folder called **temperature.html**.
- Create a new JavaScript file in this folder called **temperature.js** and link it to **temperature.html**.
- Write the code that will do the following:

    1. The user should be allowed to choose which calculation they want to do. The first output that the user sees when the program runs should say:

        ```
        In which metric is the temperature you are converting?
        C - Celsius
        F - Fahrenheit
        K - Kelvin
        ```

    2. The user should then be asked to input the number they want to convert.
    3. Next, the user should choose what they want to convert their current temperature to. For example:

        ```
        To which metric would you like to convert the temperature?
        C - Celsius
        F - Fahrenheit
        K - Kelvin
        ```

    4. The program should print out the answer in the following format: `-273°C is 0K.`
    5. The program should be able to convert from any of the three metrics to any of the three metrics.

<u>Conversion formulae:</u>

- **Celsius from:**
    - Fahrenheit: C = (F -32) x 5/9

- Kelvin: C = K - 273.15

- **Fahrenheit from:**
  - Celsius: F = C x 9/5 + 32
  - Kelvin: F = k x 9/5 - 459.67

- **Kelvin from:**
  - Celsius: K = C + 273.15
  - Fahrenheit: K = (F + 459.67) x 5/9

Rate us

Share your thoughts

HyperionDev strives to provide internationally-excellent course content that helps you achieve your learning outcomes.

Think that the content of this task, or this course as a whole, can be improved, or think we've done a good job?

Click here to share your thoughts anonymously.