|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| WEB\_10\_자바스크립트1 | 학번 : | 20215247 | 이름 : | 조아현 |

* 개념 확인

1. 괄호 안을 채워 넣으시오
2. 자바 스크립트 객체는 키와 값으로 구성된 ( 프로퍼티 )들의 집합이다
3. 자바 스크립트 객체의 프로퍼티 값이 함수일 경우 일반 함수와 구분하기 위해 ( 메소드 )라고 부른다
4. 자바 스크립트 객체의 프로퍼티 키는 빈 문자열을 포함하는 모든 ( 문자열 )또는 심볼값을 사용한다
5. 프로퍼티 또는 메소드명 앞에 작성하는 ( this )는 생성자 함수가 생성할 인스턴스를 의미한다
6. 생성자 함수를 사용한 객체 생성시 ( new ) 키워드를 사용한다
7. 프로퍼티 값을 읽기 위해 대괄호 표기법을 사용할 경우 대괄호 내에 들어가는 프로퍼티 키는 반드시 ( 문자열 ) 이어야 한다
8. 생성자 함수 프로토타입을 사용할 경우 내부에는 ( 프로퍼티 )만 존재한다.
9. 클래스에서 인스턴스 프로퍼티는 반드시 ( constructor ) 에 정의되어야 한다
10. 객체 내에 특정 프로퍼티 존재 여부를 확인하려면 ( in )연산자를 사용한다
11. ( 생성자 함수 )로 객체를 생성할 경우 반드시 new 연산자가 있어야 한다
12. 리터럴 표기법으로 book 객체를 생성하는 문장을 선택하시오 ①
13. let book={title:'js', price:30000}
14. let book={title='js', price=3000}
15. let book={title='js'; price=3000}
16. let book=[title:'js', price:30000]
17. 2번에서 생성된 book 객체에 접근하는 방법을 모두 선택하시오 ②, ④
18. book[title]
19. book.title
20. book->title
21. book['title']
22. 생성자 함수를 사용하여 객체를 정의하는 문장을 선택하시오 ②
23. let Book = function(title, price){

this.title=title; this.price=price;

}

1. function Book(title, price){

this.title=title; this.price=price;

}

1. let Book = (title, price) => {

this.title=title; this.price=price;

}

1. function Book(title, price){

this.title=title; this.price=price;

}

Book.prototype.total=title;

1. 4번의 생성자 함수를 사용하여 객체를 생성하는 문장을 제시하시오. 단, 매개값은 임의로 정할 것

==풀이==

function Book(title, price){

this.title=title; this.price=price;

}

const book = new Book(“누구에게도 상처받을 필요는 없다”, 14000);

1. 생성자 함수와 클래스로 객체를 생성하는 경우 차이점은 무엇인가?

==풀이==

생성자 함수는 객체를 생성하면 동작하며 객체가 계속 만들어지지만, 클래스로 객체 생성을 한다면, 한번만 만들고 상속해서 사용된다.

= 생성자 함수는 new를 생략하면 일반 함수로 동작하고 new연산자를 사용하면 객체 생성 클래스는 반드시 new 연산자로 객체를 생성하며 생략할 수 없다.

1. 질문에 답하시오
2. Object 생성자 함수를 사용하여 빈 객체를 생성하는 문장을 제시하시오. 단 객체명은 obj1

==풀이==

let obj1=new Object();

1. 1에서 생성된 객체에 다음과 같은 프로퍼티를 추가하고 임의의 값으로 초기화 한다.

time(자료타입 number), message(자료타입 string)

==풀이==

obj1.time=0;

obj1.message=’Object’;

1. console.log(age in obj1); 실행 결과를 제시하시오.

==풀이==

Uncaught ReferenceError: age is not defined

= in은 존재 여부 파악

Console.log(‘age’ in obj1); => false 따옴표

1. 객체 생성과 메소드 호출을 참고하여 Book class를 작성하시오

const book = new Book('흑산', '김훈');

book.bwrite(); //객체 프로퍼티 값을 웹브라우저로 출력

==풀이==

      class Book{

            constructor(title, name){

                this.title = title;

                this.name = name;

            }

            bwrite(){

                return document.write(`제목 : ${this.title}, 저자 : ${this.name}`);

            }

        }

1. 8에서 생성된 객체의 모든 프로퍼티를 순회하면서 출력하는 문장을 작성하시오. 힌트)for~in

==풀이==

      for (key in book) {

            document.write(`${key}: ${book[key]}<br>`);

        }

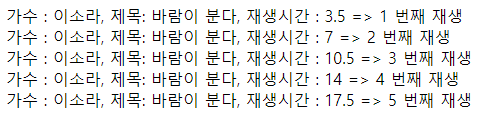
Const key

* **개념 활용 응용 프로그래밍**

1. 다음과 같은 속성과 메소드로 구성되는 객체를 제시된 방법으로 생성하고 결과를 확인하세요

* 속성 : 가수 이름, 곡명, 재생시간
* 메소드 : play(cnt) – cnt 횟수만큼 반복 재생
* 객체 생성 방법

- 객체 리터럴



|  |
| --- |
| [소스]      <script>          const song = {              name : '이소라',              title : '바람이 분다',              time : 3.5,              play : function (cnt) {                  for( let i = 1 ; i <= cnt ; i++){                      document.write( `가수: ${song.name}, 제목: ${song.title}, 재생시간: ${song.time \* i} => ${i} 번째 재생<br>`);                      }                  }          };            new song.play(5);      </script> |
| [실행 결과] |

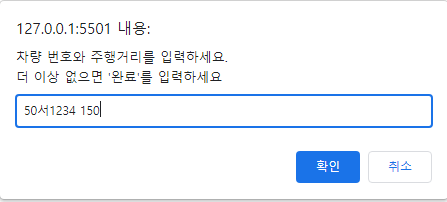
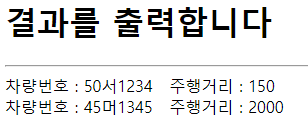
1. 다음과 같은 속성과 메소드로 구성되는 객체를 생성하는 프로그램을 생성자 함수 프로토타입을 사용하여 구현한 후 제시된 결과처럼 동작할 수 있도록 프로그램을 작성하시오

– 속성 : 차량번호, 주행거리

– 메소드 : 주행거리를 dist 만큼 증가시키는 addMileage(dist) 메소드, 반환값 없음

차량번호와 주행거리를 문자열로 반환하는 toString()

주행거리 매개변수 만큼 증가?

힌트1) 데이터 입력은 prompt()함수를 사용하고 차량번호와 주행거리는 공백으로 구분한다

힌트2) 입력된 데이터는 split() 함수를 사용하여 구분한 후 객체 초기화에 사용한다

힌트3) 초기화된 객체는 Array에 저장한다.

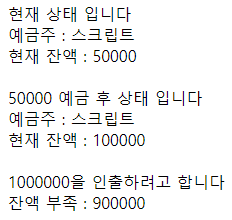
|  |
| --- |
| [소스]      <script>          function Car(...args) {              this.num = [];              this.drive = [];              for (let i = 0; i < args.length; i++) {                  if (i % 2 == 0) {                      this.num.push(args[i]);                  }                  else {                      this.drive.push(args[i]);                  }              }          }          Car.prototype.addMileage = function (dist) {              for (let i = 0; i < this.drive.length; i++) {                  this.drive[i] = Number(this.drive[i]) + dist;              }          }          Car.prototype.toString = function () {              let str = `<h1>결과를 출력합니다</h1> <hr>`;              for (let i = 0; i < this.num.length; i++) {                  str += (`차량번호 : ${this.num[i]}  주행거리 : ${this.drive[i]}<br>`);              }              return str;          }          let array = [];          let c1 = '';          while (true) {              c1 = prompt(`차량 번호와 주행거리를 입력하세요\n더 이상 없으면 완료를 입력하세요`, []);              if (c1 == '완료') break;              let c1\_arr = c1.split(" ");              array.push(...c1\_arr);          }          let car = new Car(...array);          document.write(car.toString());          car.addMileage(100);          document.write(car.toString());      </script>  For(items of array){  Document.write(${items.toString()})  }  &nbsp; |
| [실행 결과] |

1. 다음과 같은 속성과 메소드로 구성되는 클래스 Account를 만들고 제시된 결과처럼 실행되는 프로그램을 작성하세요.
   * 속성 : 예금주, 잔액
   * 메소드

- 매개변수로 받은 값 만큼 잔액을 증가하는 deposit(매개변수) 메소드, 반환값 없음

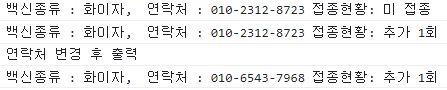
- 매개변수로 받은 값 만큼 잔액을 감소하는 withdraw(매개변수) 메소드, 반환값 없으며 잔액이 적으면 “잔액부족” 출력

- 예금주와 잔액을 출력하는 display() 메소드, 매개변수 없음



|  |
| --- |
| [소스]      <script>          class Account{              constructor(name, money){                  this.name = name;                  this.money = money;              }              deposit(dep){                  document.write(`${dep} 예금 후 상태 입니다<br>`);                  this.money += dep;              }              withdraw(wit){                  document.write(`${wit}을 인출하려고 합니다<br>`);                  if(this.money - wit < 0){                      return document.write(`잔액부족 : ${wit-this.money}`);                  }                  else{                      this.money -= wit;                  }              }              display(){                  return document.write(`                  예금주 : ${this.name}<br>                  현재 잔액 : ${this.money}<br>                  <br> `);              }          }          const acc = new Account('스크립트', 50000);          document.write(`현재 상태 입니다<br>`);          acc.display();          acc.deposit(50000);          acc.display();          acc.withdraw(1000000);      </script> |
| [실행 결과] |

1. 다음과 같은 속성과 동작을 갖는 대상을 자바스크립트 객체로 구현하고 테스트 하시오. 단, 클래스로 구현하고 테스트 결과는 console.log()를 사용하여 처리하시오.

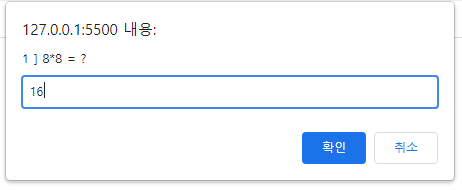


|  |  |
| --- | --- |
| 속성 | 값 |
| 백신 | 모더나, 화이자 (두개 중 하나) |
| 접종 횟수 | 0 |
| 연락처 | 010-2193-5234 |
| 동작 | 내용 |
| isFinished() | 접종 횟수가 2이면 ‘접종 완료’, 1이면 ‘추가 1회’, 0이면 ‘미 접종’ 반환 |
| addShot() | 접종 회수를 +1 증가, 만약 접종 회수가 2이면 증가 없음 |
| changeTel(value) | 연락처를 value값으로 변경 |

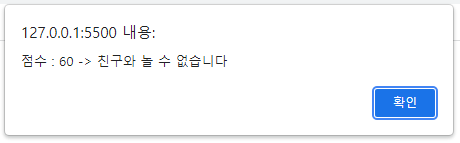
|  |
| --- |
| [소스]      <script>          class Vaccine{              constructor(vaccine, times, tel){                  this.vaccine = vaccine;                  this.times = times;                  this.tel = tel;              }              isFinished(){                  //접종 횟수가 2이면 ‘접종 완료’, 1이면 ‘추가 1회’, 0이면 ‘미 접종’ 반환                  if(this.times == 2) return '접종 완료';                  else if(this.times == 1) return '추가 1회';                  else if(this.times == 0) return '미 접종';              }                addShot(){                  //접종 회수를 +1 증가, 만약 접종 회수가 2이면 증가 없음                  if(this.times == 2) return;                  else this.times += 1;              }              changeTel(value){                  //연락처를 value값으로 변경                  console.log('연락처 변경 후 출력');                  this.tel = value;              }          }          const v1 = new Vaccine('화이자', 0, '010-2193-5234');          console.log(`백신종류 : ${v1.vaccine}, 연락처 : ${v1.tel}, 접종현황 : ${v1.isFinished()}`);          v1.addShot();          console.log(`백신종류 : ${v1.vaccine}, 연락처 : ${v1.tel}, 접종현황 : ${v1.isFinished()}`);          v1.changeTel('010-6543-7968')          console.log(`백신종류 : ${v1.vaccine}, 연락처 : ${v1.tel}, 접종현황 : ${v1.isFinished()}`);      </script> |
| [실행 결과] |

1. 2학년 조카의 구구단 학습 도우미 프로그램을 제시된 결과처럼 실행되도록 프로그램하세요.

* 1~9사이에 생성된 난수를 입력창에 제시된 결과처럼 출력하고, 답을 입력 받는다(10번 반복)



* 맞춘 회수에 10을 곱하여 점수를 계산한다.
* 계산된 점수가 90이상이면 ‘친구와 놀아도 됩니다’, 80 이상이면 ‘한번 더 연습하세요’, 70 이상이면 ‘두번 더 연습하세요’, 70미만이면 ‘친구와 놀 수 없습니다’를 알림창으로 출력



* Gugudan 클래스를 정의하여 사용하도록 한다.
* 랜덤 난수 사용

|  |
| --- |
| [소스]      <script>          class Gugudan {              constructor() {              }              score(test){                  if(test >= 90) return '친구와 놀아도 됩니다';                  else if(test >= 80) return '한번 더 연습하세요';                  else if(test >= 70) return '두번 더 연습하세요';                  else if(test < 70) return '친구와 놀 수 없습니다';              }              test() {                  let sum = 0;                  for (let i = 1; i <= 10; i++) {                      let a = parseInt(Math.random() \* (9 - 1 + 1)) + 1;                      let b = parseInt(Math.random() \* (9 - 1 + 1)) + 1;                      let answer = prompt(`${i}] ${a} \* ${b} = ?`,0);                      if(answer == (a\*b)) sum += 1;                  }                    alert(`점수 : ${sum\*10} -> ${this.score(sum\*10)}`);              }          }          let gugu = new Gugudan();          gugu.test();        </script>  = 자료형과 값 둘 다 정확하게 같은 지 비교할 떄는 === 3번 써준다.  Or Number(answer) == a\*b |
| [실행 결과] |