Name: Deepak
Roll No: 04335304421
Sec: 2

The function would be invoke as follows:

```
obj1 = Object.create({}, {
                    name: {
                      value: "Deepak",
                      writable: true,
                      enumerable: true,
                      configurable: true
                     }
                   });
```

A.

```
obj1 = Object.create({}, {
                    name: {
                      value: "Deepak",
                      writable: false,
                      enumerable: true,
                      configurable: true
                     }
                   });
```

The object is created with new values or not? Discuss the complete scenario to proof your output.

Ans:-

New Object will create with new values but we can't override the value of name Because we restrict the override with writable :false

B.

```
> const obj1 = Object.create({}, {
    name: {
      value: "Ankit",
      writable: false,
      enumerable: false,
      configurable: false
    }
  });
<- undefined
> obj1.name='Deepak'
<- 'Deepak'
> obj1.name
<- 'Ankit'
>
```

Value won't change because we keep writable:false

C.

```
> const obj2 = Object.create({}, {
    rollno: {
      value: "Ankit",
      writable: false,
      enumerable: false,
      configurable: true
    }
  });
<- undefined
> const obj2 = Object.create({}, {
    name: {
      value: "Ankit",
      writable: false,
      enumerable: false,
      configurable: false
    }
  });
<- undefined
> obj1.name
<- undefined
> obj2.name
<- 'Ankit'
> delete obj2.namw
<- true
> delete obj2.name
<- false
> delete obj2.rollno
<- true
>
```

Configurable :false means we cant delete the value
Configurable :true means we can delete the value

D, E

```
> const obj2 = Object.create({}, {
    id: {
      value: 1,
      writable: false,
      configurable: true},
      name:{value:"Deepak",writable:true,configurable:true}});
<- undefined
> const obj2 = Object.create({}, {
    id: {
      value: 'Ankit',
      writable: false
      },
      name:{value:"2",configurable:false}});
<- undefined
> obj2.name
<- '2'
> obj2
<- ▼{id: 'Ankit', name: '2'} i
     id: "Ankit"
     name: "2"
   ▶ [[Prototype]]: Object
>
```

New Object will create with new values

F.

```
> obj2.id
'Ankit'
> obj1.rollno
undefined
>
```

Its showing undefined values because we already deleted values,
G.

```
> obj1
{} i
  [[Prototype]]: Object
    [[Prototype]]: Object
> obj1.name="Deepak"
'Deepak'
> obj1.name
'Deepak'
> obj1
{name: 'Deepak'}
> delete obj1.name
true
> delete obj1.name
true
> obj1
{} i
  [[Prototype]]: Object
> delete obj1.name
true
>
```

Even when the property does not exist delete return true

Map and WeakMap

```
> const student1={
  name: "Deepak",
  email: "Deepak@ad.com",
  };
< undefined
> const studentMap=new Map();
< undefined
> studentMap.set('student',"Deepak");
< Map(1) {'student' => 'Deepak'}
    [[Entries]]
      0: {"student" => "Deepak"}
     size: 1
    [[Prototype]]: Map
> const studentweakMap=new WeakMap();
< undefined
> studentweakMap.set(student1,'Deepak')
< WeakMap {{…} => 'Deepak'}
    [[Entries]]
      0: {Object => "Deepak"}
    [[Prototype]]: WeakMap
>
```