

# Function Call Operator Solutions

- What is the prototype of the function call operator?

`some_type operator ()(...);` // Can take any number of arguments

- How is it invoked?

- To invoke the function call operator, do

`a(...);`

- How is it called?

- It will be called as

`a.operator ()(...);`

- What restrictions apply to a programmer writing an overloaded function call operator?
  - Very few
  - The operator can do anything that a member function can do
  - It can take any number and type of arguments that the programmer wishes
  - It can return any given type

- Explain what a functor is
  - In C++, a functor is a class with an overloaded function call operator
- What is a functor used for?
  - It is a class that behaves like a callable function
  - Makes the function into a “first class object” (can be passed to and returned from calls to other functions)

- Explain what the class below does
  - It has a function call operator which returns true if its argument is even, otherwise false
- Implement this class and write a program to exercise it

```
class evenp {  
    public:  
        bool operator() (int n) {  
            return (n % 2 == 0);  
        }  
};
```

- Describe what the code below does
  - Iterates over every element in v
  - Calls evenp's function call operator on the current element
  - If it returns true, displays a message
- Why is the second argument passed by value?
  - evenp has no data members, so copying it is more efficient than going through a pointer or reference

- Write a full working program which exercises this code

```
void do_it(const vector<int>& vec, evenp is_even) {  
    for (auto v: vec)  
        if (is_even(v))  
            cout << v << " is even\n";  
}
```

- Explain what it means when a functor has state
  - It means the class has data members which can store data between successive calls to the function call operator



- Explain what the class below does
  - It has a private data member, which has a default value of 1
  - (This avoids a potential division by 0 if the constructor does not set it)
  - This is initialized in the constructor
  - Every time operator() is called, its argument is modulo divided by this member
  - The operator returns true only if there is no remainder, i.e the argument is exactly divisible

- Explain what the class below does
- Implement this class and write a program to exercise it

```
class divisible {  
    private:  
        int divisor {1};  
    public:  
        divisible(int d) : divisor(d) {}  
        bool operator() (int n) {  
            return (n % divisor == 0);  
        }  
};
```