**Declare interface:**

type AirthOperation interface{

Add() float32

Subtract(float32, float32) float32

Multiply(int, int) int

}

func main() {

type AirthOperation interface{

Add() float32

Subtract(float32, float32) float32

Multiply(int, int) int

}

var interface\_var AirthOperation

fmt.Println(interface\_var) // <nil>

}

**Implement interface:**

package main

import "fmt"

// step 1

type AirthOperation interface{

Add() float32

Subtract(float32, float32) float32

Multiply(int, int) int

}

// step 2

type Operandset1 struct{

op1 float32

op2 float32

}

//step 3

func (operandset1\_var Operandset1) Add() float32 {

return operandset1\_var.op1 + operandset1\_var.op2

}

// step 6

type Operandset2 struct{

op1 int

op2 int

}

// step 7

func (operandset2\_var Operandset2) Add(a int, b int) float32 {

return float32(a + b)

}

func main() {

// step 4

operandset1\_var := Operandset1{3.6, 6.9}

// step 8

operandset2\_var := Operandset2{45, 67}

// step 5

fmt.Println(operandset1\_var.Add())

// step 9

fmt.Println(operandset2\_var.Add(operandset2\_var.op1, operandset2\_var.op2))

}

package main

import "fmt"

// step 1

type AirthOperation interface {

Add() float32

Subtract(float32, float32) float32

Multiply(int, int) int

}

// step 2

type Operandset1 struct {

op1 float32

op2 float32

}

// step 3

type Operandset2 struct {

op1 int

op2 int

}

// step 4 - Define all abstract methods

func (operandset1\_var Operandset1) Add() float32 {

return operandset1\_var.op1 + operandset1\_var.op2

}

func (operandset1\_var Operandset1) Subtract(v1 float32, v2 float32) float32 {

fmt.Println("Passed values in Subtract method: ", v1, v2)

return operandset1\_var.op1 - operandset1\_var.op2

}

func (operandset2\_var Operandset2) Multiply(v1 int, v2 int) int {

fmt.Println("Passed values in Multiply method: ", v1, v2)

return operandset2\_var.op1 \* operandset2\_var.op2

}

func main() {

// step 5

a := Operandset1{3.6, 6.9}

b := Operandset2{3, 6}

// step 6

fmt.Println("Return value from Add method with Operandset1: ", a.Add())

fmt.Println("Return value from Subtract method with Operandset1: ", a.Subtract(2, 3))

fmt.Println("Return value from Multiply method with Operandset2: ", b.Multiply(2, 3))

}

**Empty interface:**

package main

import "fmt"

type empty\_interface interface{

}

func show\_value(interface\_var empty\_interface){

fmt.Println(interface\_var)

}

func main() {

type string\_struct struct{

a string

}

string\_struct\_var := string\_struct{"Hello world"}

type float\_struct struct{

a float32

}

float\_struct\_var := float\_struct{77.88}

show\_value(string\_struct\_var) // {Hello world}

show\_value(float\_struct\_var) // {77.88}

}

**Implement multiple interface:**

package main

import "fmt"

type AirthOperation interface{

Add() float32

}

type CompareOperation interface{

Compare(float32, float32) bool

}

type Operandset1 struct{

op1 float32

op2 float32

}

func (operandset1\_var Operandset1) Add() float32 {

return operandset1\_var.op1 + operandset1\_var.op2

}

func (operandset1\_var Operandset1) Compare(a float32, b float32) bool {

return a==b

}

func main() {

operandset1\_var := Operandset1{3.6, 6.9}

fmt.Println(operandset1\_var.Add()) // 10.5

fmt.Println(operandset1\_var.Compare(operandset1\_var.op1, operandset1\_var.op2)) // false

var AirthOperation\_var AirthOperation = operandset1\_var

fmt.Println(AirthOperation\_var.Add()) // 10.5

var compareOperation\_var CompareOperation = operandset1\_var

fmt.Println(compareOperation\_var.Compare(operandset1\_var.op1, operandset1\_var.op2)) // false

}

**Type assertion:**

package main

import "fmt"

type AirthOperation interface{

Add() float32

}

type Operandset1 struct{

op1 float32

op2 float32

}

type Operandset2 struct{

op1 float32

op2 float32

}

func (operandset1\_var Operandset1) Add() float32 {

return operandset1\_var.op1 + operandset1\_var.op2

}

func main() {

var interface\_var AirthOperation

interface\_var = Operandset1{3.6, 6.9}

t, ok := interface\_var.(Operandset1)

fmt.Println(t, ok) // {3.6 6.9} true

}

**Type switch:**

package main

import "fmt"

type AirthOperation interface{

Add() float32

}

type Operandset1 struct{

op1 float32

op2 float32

}

type Operandset2 struct{

op1 int

op2 int

}

type Operandset3 struct{

op1 string

op2 string

}

func main() {

check(Operandset1{3.7, 6.7})

check(Operandset2{37, 67})

check(Operandset3{"abc", "def"})

}

func check(i interface{}) {

switch v := i.(type) {

case Operandset1:

fmt.Println("Interface has a value of type Operandset1 and the value is", v)

case Operandset2:

fmt.Println("Interface has a value of type Operandset1 and the value is", v)

default:

fmt.Println("Default case, Interface doesn't have a value of type", v)

}

}

**Embedding interface:**

package main

import "fmt"

type AirthOperation interface{

Add() float32

}

type CompareOperation interface{

Compare() bool

}

type Operation interface{

AirthOperation

Compare() bool

}

type Operandset1 struct{

op1 float32

op2 float32

}

func (operandset1\_var Operandset1) Add() float32 {

return operandset1\_var.op1 + operandset1\_var.op2

}

func (operandset1\_var Operandset1) Compare() bool {

return operandset1\_var.op1 == operandset1\_var.op2

}

func main() {

var Operation\_var Operation

var CompareOperation\_var CompareOperation

var AirthOperation\_var AirthOperation

Operation\_var = Operandset1{5.78, 6.8}

fmt.Println(Operation\_var, Operation\_var.Add(), Operation\_var.Compare())

CompareOperation\_var = Operandset1{6.8, 6.8}

fmt.Println(CompareOperation\_var, CompareOperation\_var.Compare())

AirthOperation\_var = Operandset1{5, 7.8}

fmt.Println(AirthOperation\_var, AirthOperation\_var.Add())

}

**Interfaces as polymorphism:**

package main

import "fmt"

type AirthOperation interface{

Add() float32

}

type Operandset1 struct{

op1 float32

op2 float32

}

type Operandset2 struct{

op1 int

op2 int

}

func (operandset1\_var Operandset1) Add() float32 {

return operandset1\_var.op1 + operandset1\_var.op2

}

func (operandset2\_var Operandset2) Add() float32 {

return float32(operandset2\_var.op1 + operandset2\_var.op2)

}

func main() {

var interface\_var AirthOperation

interface\_var = Operandset1{5.78, 6.8}

fmt.Println(interface\_var.Add()) // 12.58

interface\_var = Operandset2{5, 6}

fmt.Println(interface\_var.Add()) //11

}