```
package main

import  (
      "fmt"
      "strings"
      "unsafe"
)

type (
      stringHeader struct {
            start   uintptr
            length  uint
      }
)

func main() {
    str:= strings.Repeat("abc",4)
    fmt.Println(str)                                    // abcabcabcabc
    fmt.Println(&str)                                   // 0xc00002c1f0
    fmt.Println(unsafe.Pointer(&str))                   // 0xc00002c1f0
    fmt.Println(unsafe.Sizeof(str))                     // 16
    strarr := (*[2]uintptr)(unsafe.Pointer(&str))
    fmt.Println(*strarr)                                // [824633778352 12]
    fmt.Printf("%x %d\n", (*strarr)[0], (*strarr)[1])   // c00000e0b0 12
    // all together - struct
    fmt.Println(*(*stringHeader)(unsafe.Pointer(&str))) // {824633778352 12}
    str2:= str
    fmt.Println(&str2)                                  // 0xc00002c210
    fmt.Println(*(*stringHeader)(unsafe.Pointer(&str2)))// {824633778352 12}
}
```

01a.go

```go
package main

import  (
    "fmt"
    "unsafe"
)

type (
    stringHeader struct {
        start   unsafe.Pointer
        length  uint
    }
)

func getString(s string) *stringHeader {
    return (*stringHeader)(unsafe.Pointer(&s))
}

func main() {
    str:= "ABCDEFG"
    fmt.Println(str)       // ABCDEFG
    fmt.Println(&str)      // 0xc00003c1c0
    sh:= getString(str)
    fmt.Println(*sh)       // {0x4bf578 7}
    b:= (*byte)(unsafe.Pointer(uintptr((*sh).start)+4))
    fmt.Println(&b, b)     // 0xc000070020 0x4bf57c
    *b = '+'
    // unexpected fault address 0x4bf57c
    // fatal error: fault
    // ...
    fmt.Println(str)
}
```

01b.go

```go
package main

import  (
     "fmt"
     "unsafe"
)

type (
    stringHeader struct {
        start   unsafe.Pointer
        length  uint
    }
)

func getString(s string) *stringHeader {
    return (*stringHeader)(unsafe.Pointer(&s))
}

func main() {
    str:= "ABCDEFG"
    fmt.Println(str)       // ABCDEFG
    fmt.Println(&str)      // 0xc00003c1c0
    sh:= getString(str)
    fmt.Println(*sh)       // {0x4bf579 7}

    str1:= "ABCDEFG"
    fmt.Println(str1)      // ABCDEFG
    fmt.Println(&str1)     // 0xc00003c1f0
    sh1:= getString(str1)
    fmt.Println(*sh1)      // {0x4bf579 7}

    str1 += "!"
    fmt.Println(str1)      // ABCDEFG!
    fmt.Println(&str1)     // 0xc00003c1f0
    sh1 = getString(str1)
    fmt.Println(*sh1)      // {0xc0000480b0 8}
    b:= (*byte)(unsafe.Pointer(uintptr((*sh1).start)+4))
    fmt.Println(&b, b)     // 0xc000070020 0xc0000480b4
    *b = '+'
    fmt.Println(str1)      // ABCD+FG!
}
```

01c.go

```go
package main

import  (
      "fmt"
      "unsafe"
)

type (
      Slice struct {
         start   unsafe.Pointer
         len     int
         cap     int
      }
)

func printSlice(s []uint)  {
    ps:= (*Slice)(unsafe.Pointer(&s))
    fmt.Println((*ps).start, (*ps).len, (*ps).cap)
}

func main() {
    var p []uint
    fmt.Println(p)              // []
    printSlice(p)               // <nil> 0 0
    p = make([]uint, 2)
    fmt.Println(p)              // [0 0]
    printSlice(p)               // 0xc0000540a0 2 2
    p = append(p, 12345)
    fmt.Println(p)              // [0 0 12345]
    printSlice(p)               // 0xc000052140 3 4
}
```

02a.go

```go
package main

import  (
      "fmt"
      "unsafe"
)

type (
      Slice struct {
         start   unsafe.Pointer
         len     int
         cap     int
      }
)

func printSlice(s []uint)  {
    ps:= (*Slice)(unsafe.Pointer(&s))
    fmt.Println((*ps).start, (*ps).len, (*ps).cap)
}

func main() {
    a:= [...]uint {0,1,2,3,4,5,6}
    fmt.Println(unsafe.Pointer(&a)) // 0xc000078040
    b:= a[2:5]
    fmt.Println(b)                  // [2 3 4]
    fmt.Println(unsafe.Pointer(&a[2])) // 0xc000078050
    printSlice(b)                   //0xc000078050 3 5
    c:= b[1:2]
    fmt.Println(c  )                // [3]
    printSlice(c)                   //0xc000078058 1 4
}
```

02b.go

```
1  package main
2
3  import "fmt"
4
5  func square(d []int) {
6       for i, x := range d {
7            d[i] = x * x
8       }
9  }
10
11 func fill(d []int) {
12      for i, _ := range d {
13           d[i] = i + 1
14      }
15 }
16
17 func main() {
18      var a [10]int
19      fmt.Println(len(a), cap(a), a) //   10 10 [0 0 0 0 0 0 0 0 0 0]
20      d := a[2:8]
21      fill(d)
22      fmt.Println(len(a), cap(a), a) //   10 10 [0 0 1 2 3 4 5 6 0 0]
23      fmt.Println(len(d), cap(d), d) //   6 8 [1 2 3 4 5 6]
24      square(a[4:7])
25      fmt.Println(len(a), cap(a), a) //   10 10 [0 0 1 2 9 16 25 6 0 0]
26      fmt.Println(len(d), cap(d), d) //   6 8 [1 2 9 16 25 6]
27 }
```

02c.go

```go
package main

import "fmt"

func main() {
      slc0:= make([]int,5)
      fmt.Println(slc0)

    // Creating slices
    slc1:= []string{"one", "two", "three", "four","five", "six", "seven", "eight"}
    var slc2 []string
    slc3:= make([]string, 5)
    slc4:= []string{"eleven", "twelve", "thirteen", "fourteen"}

    // Before copying
    fmt.Println("Slice_1:", slc1)
    fmt.Println("Slice_2:", slc2)
    fmt.Println("Slice_3:", slc3)
    fmt.Println("Slice_4:", slc4)

    // Copying the slices
    copy_1 := copy(slc2, slc1)
    fmt.Println("\nSlice:", slc2)
    fmt.Println("Total number of elements copied:", copy_1)

    copy_2 := copy(slc3, slc1)
    fmt.Println("\nSlice:", slc3)
    fmt.Println("Total number of elements copied:", copy_2)

    copy_3 := copy(slc3, slc4)
    fmt.Println("\nSlice:", slc3)
    fmt.Println("Total number of elements copied:", copy_3)

    copy_4:= copy(slc1, slc4)
    fmt.Println("\nSlice:", slc1)
    fmt.Println("Total number of elements copied:", copy_4)

    copy_5:= copy(slc1, slc1[3:])
    fmt.Println("\nSlice:", slc1)
    fmt.Println("Total number of elements copied:", copy_5)

```

```
// [0 0 0 0 0]
// Slice_1: [one two three four five six seven eight]
// Slice_2: []
// Slice_3: [    ]
// Slice_4: [eleven twelve thirteen fourteen]
//
// Slice: []
// Total number of elements copied: 0
//
// Slice: [one two three four five]
// Total number of elements copied: 5
//
// Slice: [eleven twelve thirteen fourteen five]
// Total number of elements copied: 4
//
// Slice: [eleven twelve thirteen fourteen five six seven eight]
// Total number of elements copied: 4
//
// Slice: [fourteen five six seven eight six seven eight]
// Total number of elements copied: 5
}
```

03.go

```go
package main

import  (
      "fmt"
      "unsafe"
)

type (
      Slice struct {
         start   unsafe.Pointer
         len     int
         cap     int
      }
)

func printSlice(s []int)  {
    sInfo:= *(*Slice)(unsafe.Pointer(&s))
    fmt.Println(sInfo.start, sInfo.len, sInfo.cap)
}

func main() {
      a := [...]int{1, 2, 3, 4, 5, 6, 7, 8}
      fmt.Println("a:", len(a), a)             // a: 8 [1 2 3 4 5 6 7 8]
      fmt.Println(unsafe.Pointer(&a))          // 0xc000078040
      s := a[:5]
      fmt.Println(unsafe.Pointer(&s))          // 0xc000042420
      fmt.Println("s:", len(s), cap(s), s)     // s: 5 8 [1 2 3 4 5]
      printSlice(s)                            // 0xc000078040 5 8
      s2 := s
      fmt.Println(unsafe.Pointer(&s2))         // 0xc000042460
      fmt.Println("s2:", len(s2), cap(s2), s2) // s2: 5 8 [1 2 3 4 5]
      printSlice(s2)                           // 0xc000078040 5 8
      s = append(s, -1, -2, -3, -4)            //
      fmt.Println("a:", len(a), a)             // a: 8 [1 2 3 4 5 6 7 8]
      fmt.Println("s:", len(s), cap(s), s)     // s: 9 16 [1 2 3 4 5 -1 -2 -3 -4]
      printSlice(s)                            // 0xc00007a080 9 16
      s2 = append(s2, 0)                       //
      fmt.Println("a:", len(a), a)             // a: 8 [1 2 3 4 5 0 7 8]
      fmt.Println("s2:", len(s2), cap(s2), s2) // s2: 6 8 [1 2 3 4 5 0]
      printSlice(s2)                           // 0xc000078040 6 8
      s2 = append(s2, -11, 12, 13, 14, 15)
      fmt.Println("a:", len(a), a)             // a: 8 [1 2 3 4 5 0 7 8]
```

```
	fmt.Println("s2:", len(s2), cap(s2), s2) // s2: 11 16 [1 2 3 4 5 0 -11 12 13 14 15]
	printSlice(s2)                           // 0xc00007a100 11 16
	s = a[3:6]
	fmt.Println("s:", len(s), cap(s), s)     // s: 3 5 [4 5 0]
	printSlice(s)                            // 0xc000078058 3 5
	s = append(s, 1, 2, 3, 4, 5, 6, 7, 8)
	fmt.Println("s:", len(s), cap(s), s)     // s: 11 12 [4 5 0 1 2 3 4 5 6 7 8]
	printSlice(s)                          // 0xc00003a060 11 12
	s = append(s, 1, 2, 3)
	fmt.Println("s:",len(s),cap(s),s)  // s: 14 24 [4 5 0 1 2 3 4 5 6 7 8 1 2 3]
	printSlice(s)                          // 0xc000086000 14 24
	s = s[:16]
	fmt.Println("s:",len(s),cap(s),s)  // s: 16 24 [4 5 0 1 2 3 4 5 6 7 8 1 2 3 0 0]
	printSlice(s)                          // 0xc000086000 16 24
}
```

04.go

```
package main

import "fmt"

func f(d []int) {
	for i, _ := range d {
		d[i] = i + 1
	}
	d = append(d, 1, 2, 3, 4, 5)
	fmt.Println(len(d), cap(d), d)
}

func main() {
	var a [10]int
	fmt.Println(len(a), cap(a), a)    //   10 10 [0 0 0 0 0 0 0 0 0 0]
	c := a[2:4]
	f(c)                              //   7 8 [1 2 1 2 3 4 5]
	fmt.Println(len(a), cap(a), a)    //   10 10 [0 0 1 2 1 2 3 4 5 0]
	fmt.Println(len(c), cap(c), c)    //   2 8 [1 2]
	cc := []int{11, 12, 13, 14}
	fmt.Println(len(cc), cap(cc), cc) //   4 4 [11 12 13 14]
	f(cc)                             //   9 10 [1 2 3 4 1 2 3 4 5]
	fmt.Println(len(cc), cap(cc), cc) //   4 4 [1 2 3 4]
}
```

05.go

```go
package main

import  (
    "fmt"
    "unsafe"
)

type (
     Slice struct {
        start   unsafe.Pointer
        len     int
        cap     int
     }
)

func sliceInfo(s []int) Slice  {
    return *(*Slice)(unsafe.Pointer(&s))
}

func main() {
    b:= make([]int, 0, 1)
    fmt.Println(b)          // []
    fmt.Println(sliceInfo(b))           // {0xc000048058 0 1}
    b = append(b, 7)
    fmt.Println(b)          // [7]
    fmt.Println(sliceInfo(b))           // {0xc000048058 1 1}
    b = append(b, 7)
    fmt.Println(b)          // [7 7]
    fmt.Println(sliceInfo(b))           // {0xc0000480b0 2 2}
    b = append(b, 7)
    fmt.Println(b)          // [7 7 7]
    fmt.Println(sliceInfo(b))           // {0xc0000460e0 3 4}
    fmt.Println()
    c:= cap(b)
    for i:= 0; i<1000000; i++  {
        b = append(b, i)
        if cap(b) != c  {
               fmt.Println(sliceInfo(b))
               c = cap(b)
        }
    }
```

```
// {0xc000078040 5 8}
// {0xc00007a080 9 16}
// {0xc000034100 17 32}
// {0xc000086000 33 64}
// {0xc000088000 65 128}
// {0xc00008a000 129 256}
// {0xc00008c000 257 512}
// {0xc00008e000 513 1024}
// {0xc000090000 1025 1280}
// {0xc00009a000 1281 1696}
// {0xc0000a4000 1697 2304}
// {0xc0000b6000 2305 3072}
// {0xc0000bc000 3073 4096}
// {0xc0000c4000 4097 5120}
// {0xc0000ce000 5121 7168}
// {0xc0000dc000 7169 9216}
// {0xc0000ee000 9217 12288}
// {0xc000106000 12289 15360}
// {0xc000124000 15361 19456}
// {0xc00014a000 19457 24576}
// {0xc00017a000 24577 30720}
// {0xc0001b6000 30721 38912}
// {0xc000202000 38913 49152}
// {0xc000262000 49153 61440}
// {0xc0002da000 61441 76800}
// {0xc000370000 76801 96256}
// {0xc000082000 96257 120832}
// {0xc00016e000 120833 151552}
// {0xc00044e000 151553 189440}
// {0xc0005c0000 189441 237568}
// {0xc000082000 237569 296960}
// {0xc00044e000 296961 371712}
// {0xc000082000 371713 464896}
// {0xc00044e000 464897 581632}
// {0xc0008be000 581633 727040}
// {0xc000e4a000 727041 909312}
// {0xc00044e000 909313 1136640}
}
```

06.go

```
1  package main
2
3  import "fmt"
4
5  func hello() *string {
6      s := "hello, world"
7      return &s
8  }
9
10 func main() {
11     fmt.Println(*hello())                        // hello, world
12     hi := hello()
13     fmt.Printf("Тип переменной hi %T,\n", hi)    // Тип переменной hi *string,
14     fmt.Printf("она содержит указатель на строку \"%s\"\n", *hi)
15                             // она содержит указатель на строку "hello, world"
16 }
```

07.go