Lecture 6

18 September 2018

Admin Matters
Common C Mistakes
Unit 13: **Call Stack**
Unit 14: **Pointer**
Unit 15: **Array**
Unit 16: **String**

Midterm

Venue: MPSH 1 (B)

2 October

4pm - 6pm

Midterm

Open Book

Lecture 1 to 5

Tutorial 5

Problem Sets from Units Today

Assignment 1

being graded

Assignment 1

solutions & general comments to be made available tomorrow

Assignment 2

Due this Friday 6pm

Assignment 3

Release this Friday

(to be graded on correctness and style)

Assignment 3

Everything
up to
arrays

Assignment 3

Due 5 October 2018

Practical Exam 1 Midterm

Information posted online

Previously on

CS1010..

Unit 5:

We will use long and double only for CS1010

Using `int` will cause your program to fail on large inputs.

What's the advantage of `int` over `long`?

(besides memory usage)

Using `float` will cause your program to loose precision.

What's the advantage of `float` over `double`?

(besides memory usage)

Please follow instructions given in the assignment.

e.g.,

not writing function `area_of_rectangle`

e.g.,

not solving `digits` recursively

```
long square(long x)
{
  return x*x;
}
```

```
double hypotenuse_of(long base, long height)
{
  return sqrt(square(base) + square(height));
}
```

```
double hypotenuse;
```

not
a function

```
int main()
{
  hypotenuse = hypotenuse_of(base, height);
}
```

“This is a very very bad programming habit. So, don’t do this”

There are still students who do this

Strict policy on plagiarism

Disciplinary action will be taken :(

There are still students who do this

```
bool morning_peak_hour(long hours, long minutes){
      if((hours >= 6 && hours <= 8 )|| (hours == 9 && minutes >= 0 && minutes <= 29)){
            return true;}
      else{
            return false;}
}
//function to test it boarding time falls under evening peak hour
bool evening_peak_hour(long hours, long minutes){
      if((hours >= 18 && hours <= 23) && (minutes >= 0 && minutes <= 59)){
            return true;}
      else{
            return false;}
}
//function to test if boarding time falls under midnight peak hour
bool midnight_peak_hour(long hours, long minutes){
      if((hours >= 0 && hours <= 5) && (minutes >= 0 && minutes <= 59)){
            return true;}
      else{
            return false;}
}
//function to test if it is a weekday
bool weekday(long days){
      if(days >= 1 && days <= 5){
            return true;}
      else{
            return false;}
}
//function to calculate surcharge
double surcharge(long days, long hours, long minutes){
      double surcharge = 0.0;
      if(weekday(days) == true && morning_peak_hour(hours, minutes)  == true){
            surcharge = 0.25;}
      else if(evening_peak_hour(hours, minutes) == true){
            surcharge = 0.25;}
      else if(midnight_peak_hour(hours, minutes) == true){
            surcharge = 0.50;}
      return surcharge;
}
//function to calculate basic fare
double basic_fare(distance){
      long distance1;
      long distance2;
      double charge = 0.0;

      if(distance > 0 && distance <= 1000){
            charge = 3.40;
      }
      else if(distance > 1000 && distance <= 10200){
              distance1 = distance - 1000;
              if(distance1 % 400!= 0){
                    charge = 3.40 + ((distance1)/400)*0.22 + 0.22;}
              else{
```

```
bool morning_peak_hour(long hours, long minutes){
        if((hours >= 6 && hours <= 8 )|| (hours == 9 &&  minutes >= 0 && minutes <= 29)){
                return true;}
        else{
                return false;}
}
//function to test if boarding time falls under evening peak hour
bool evening_peak_hour(long hours, long minutes){
        if((hours >= 18 && hours <= 23) && (minutes >= 0 && minutes <= 59)){
                return true;}
        else{
                return false;}
}
//function to test if boarding time falls under midnight peak hour
bool midnight_peak_hour(long hours, long minutes){
        if((hours >= 0 && hours <= 5) && (minutes >= 0 && minutes <= 59)){
                return true;}
        else{
                return false;}
}
//function to test if it is a weekday
bool weekday(long days){
        if(days >=1 && days <= 5){
                return true;}
        else{
                return false;}
}
//function to calculate surcharge
double surcharge(long days, long hours, long minutes){
        double surcharge = 0.0;
        if(weekday(days) == true && morning_peak_hour(hours, minutes)  == true){
                surcharge = 0.25;}
        else if(evening_peak_hour(hours, minutes) == true){
                surcharge = 0.25;}
        else if(midnight_peak_hour(hours, minutes) == true){
                surcharge = 0.50;}
return surcharge;
}
//function to calculate basic fare
double basic_fare(distance){
        long distance1;
        long distance2;
        double charge = 0.0;

        if(distance > 0 && distance <= 1000){
                charge = 3.40;
        }
        else if(distance > 1000 && distance <=10200){
                distance1 = distance - 1000;
                if(distance1 % 400!= 0){
                        charge = 3.40 + ((distance1)/400)*0.22 + 0.22;}
```

Call Stack

```
int main()
{
   long x = 1;
   long y;
}
```

```
int main()
{
   long x = 1;
   long y;
}
```

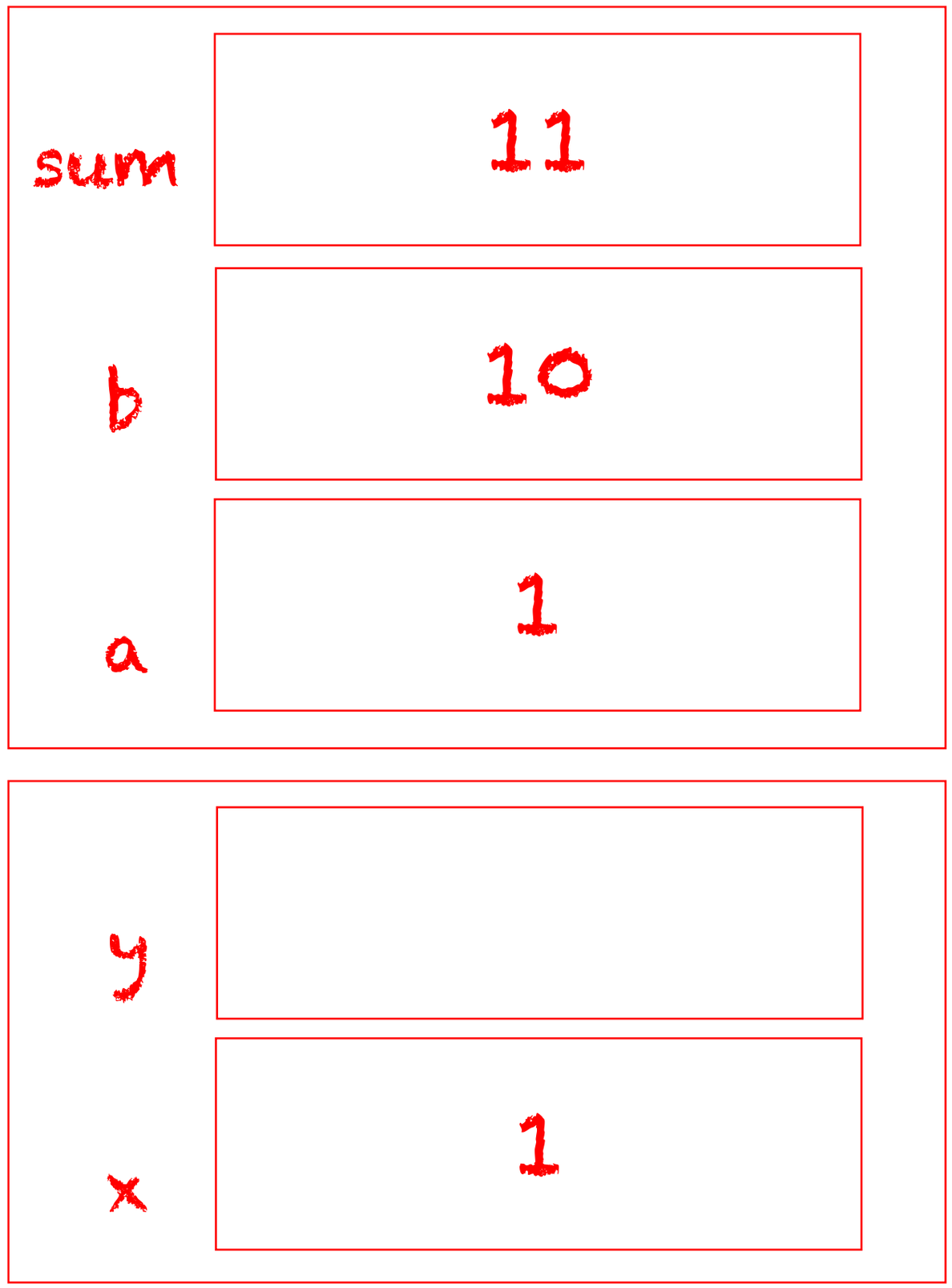
Call Stack

```
long add(long a, long b)
{
  long sum;
  sum = a + b;
  return sum;
}
```

```
int main()
{
  long x = 1;
  long y;
  y = add(x, 10);
}
```

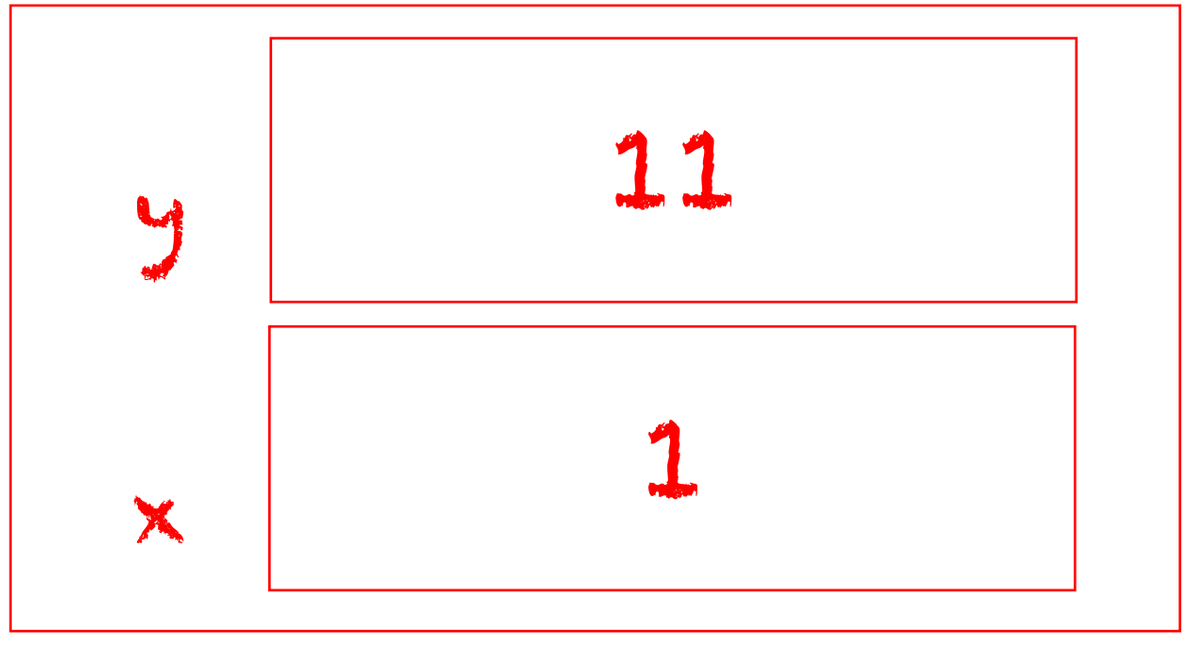
```
long add(long a, long b)
{
   long sum;
   sum = a + b;
   return sum;
}

int main()
{
   long x = 1;
   long y;
   y = add(x, 10);
}
```



```
long add(long a, long b)
{
  long sum;
  sum = a + b;
  return sum;
}
```

```
int main()
{
  long x = 1;
  long y;
  y = add(x, 10);
}
```

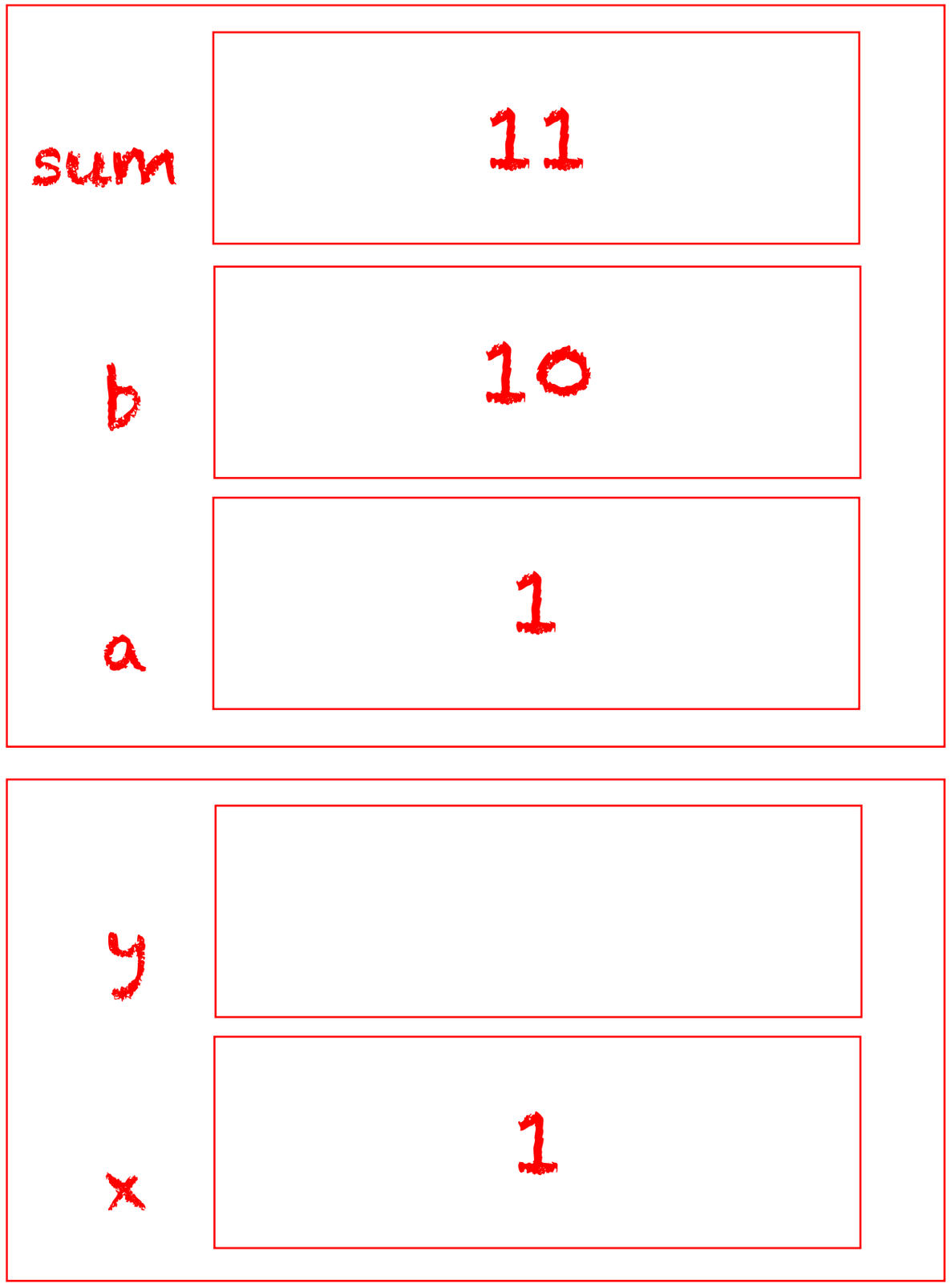


```
long add(long a, long b)
{
  long sum;
  sum = a + b;
  a = 42;
  return sum;
}

int main()
{
  long x = 1;
  long y;
  y = add(x, 10);
}
```

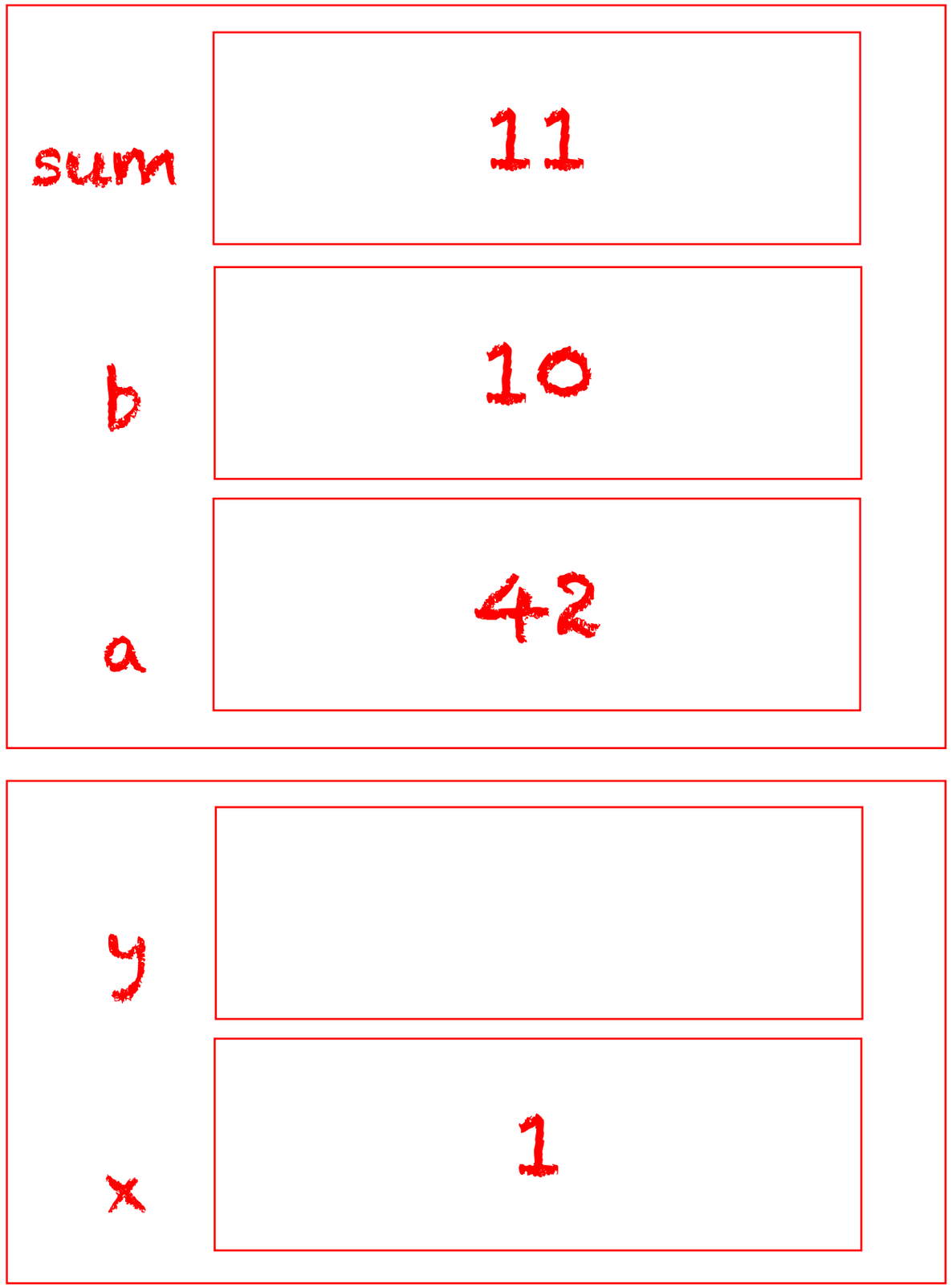
```
long add(long a, long b)
{
  long sum;
  sum = a + b;
  a = 42;
  return sum;
}

int main()
{
  long x = 1;
  long y;
  y = add(x, 10);
}
```



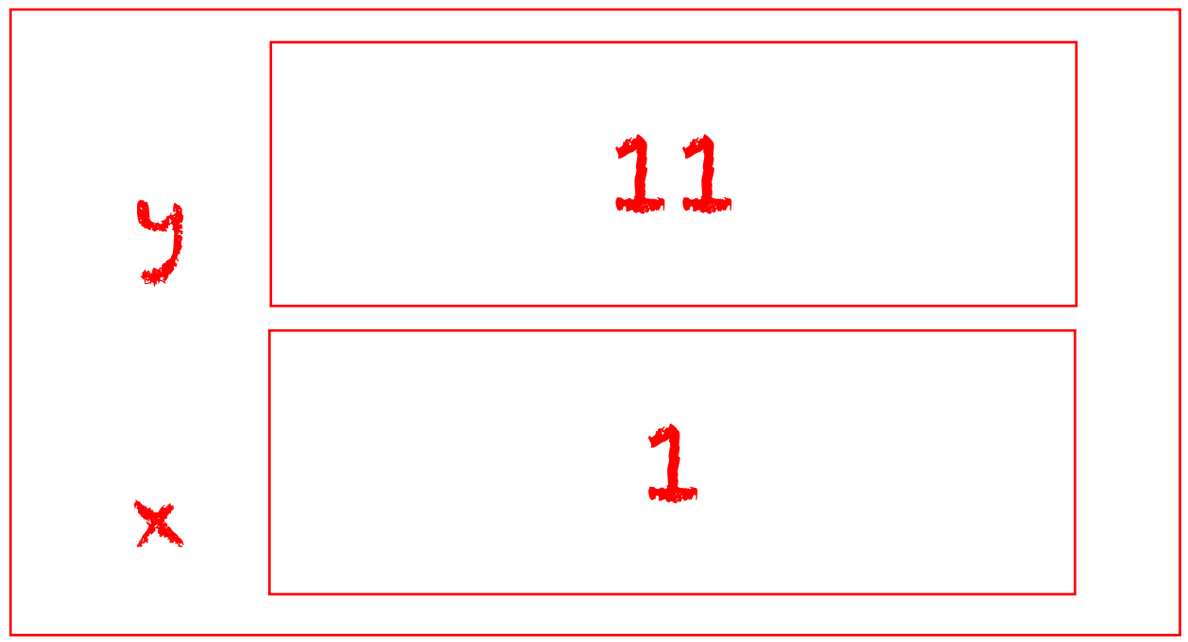
```
long add(long a, long b)
{
  long sum;
  sum = a + b;
  a = 42;
  return sum;
}

int main()
{
  long x = 1;
  long y;
  y = add(x, 10);
}
```



```
long add(long a, long b)
{
  long sum;
  sum = a + b;
  a = 42;
  return sum;
}
```

```
int main()
{
  long x = 1;
  long y;
  y = add(x, 10);
}
```

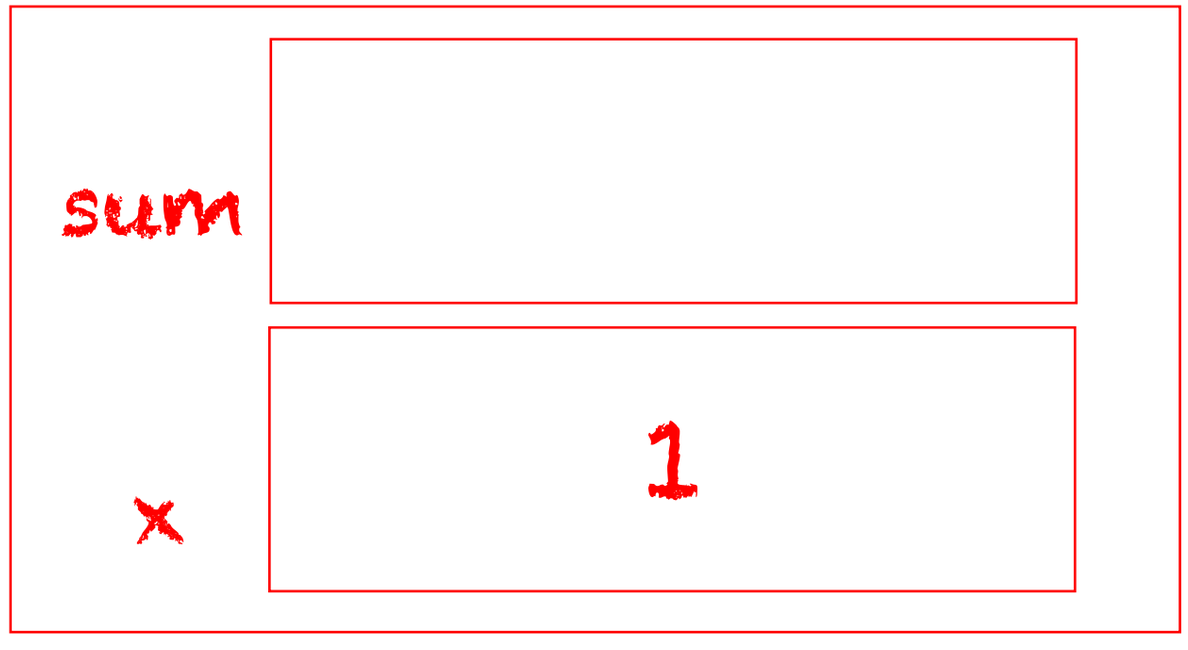


```
void add(long sum, long a, long b)
{
  sum = a + b;
}

int main()
{
  long x = 1;
  long sum;
  add(sum, x, 10);
}
```

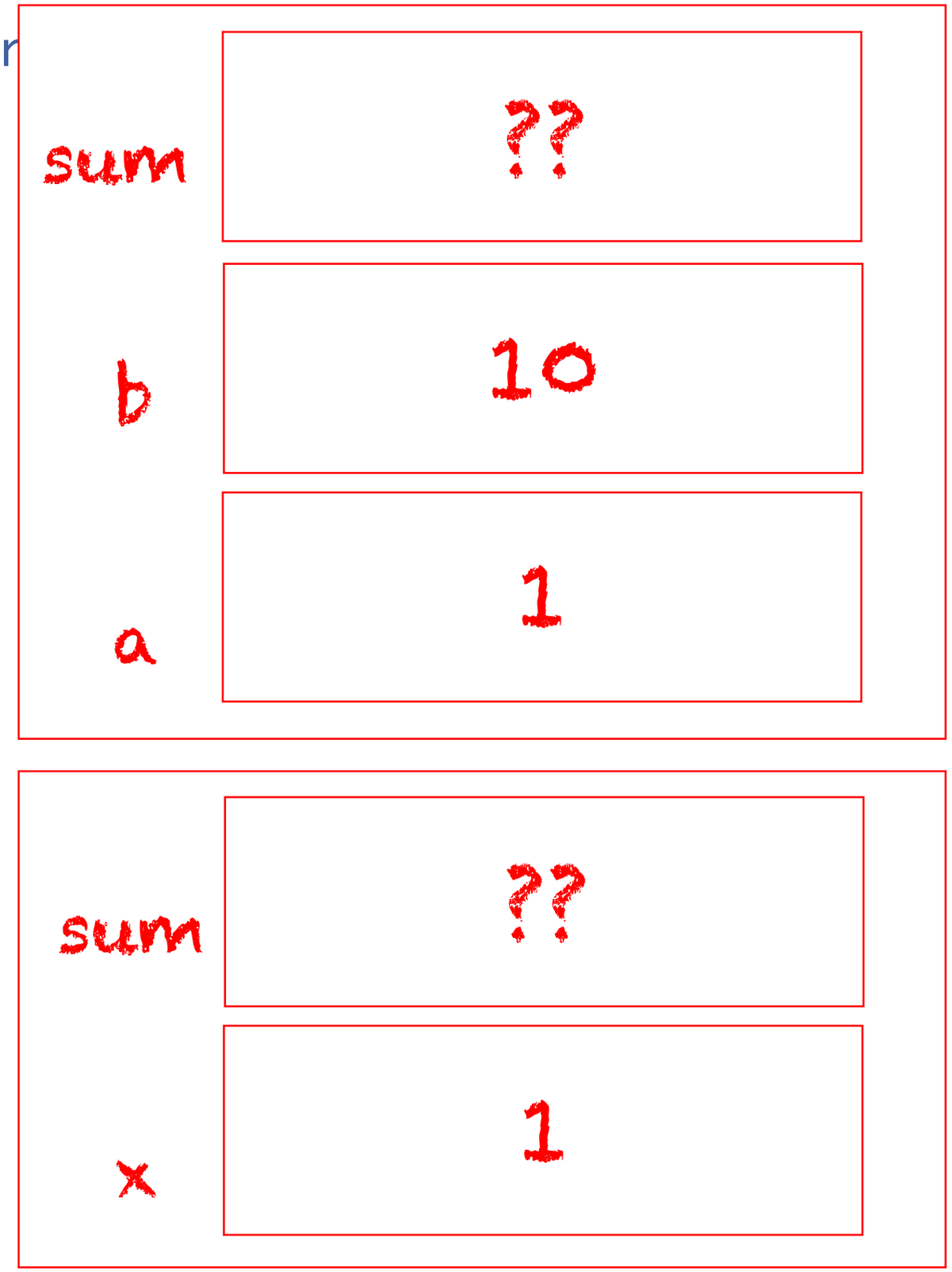
```
void add(long sum, long a, long b)
{
  sum = a + b;
}

int main()
{
  long x = 1;
  long sum;
  add(sum, x, 10);
}
```



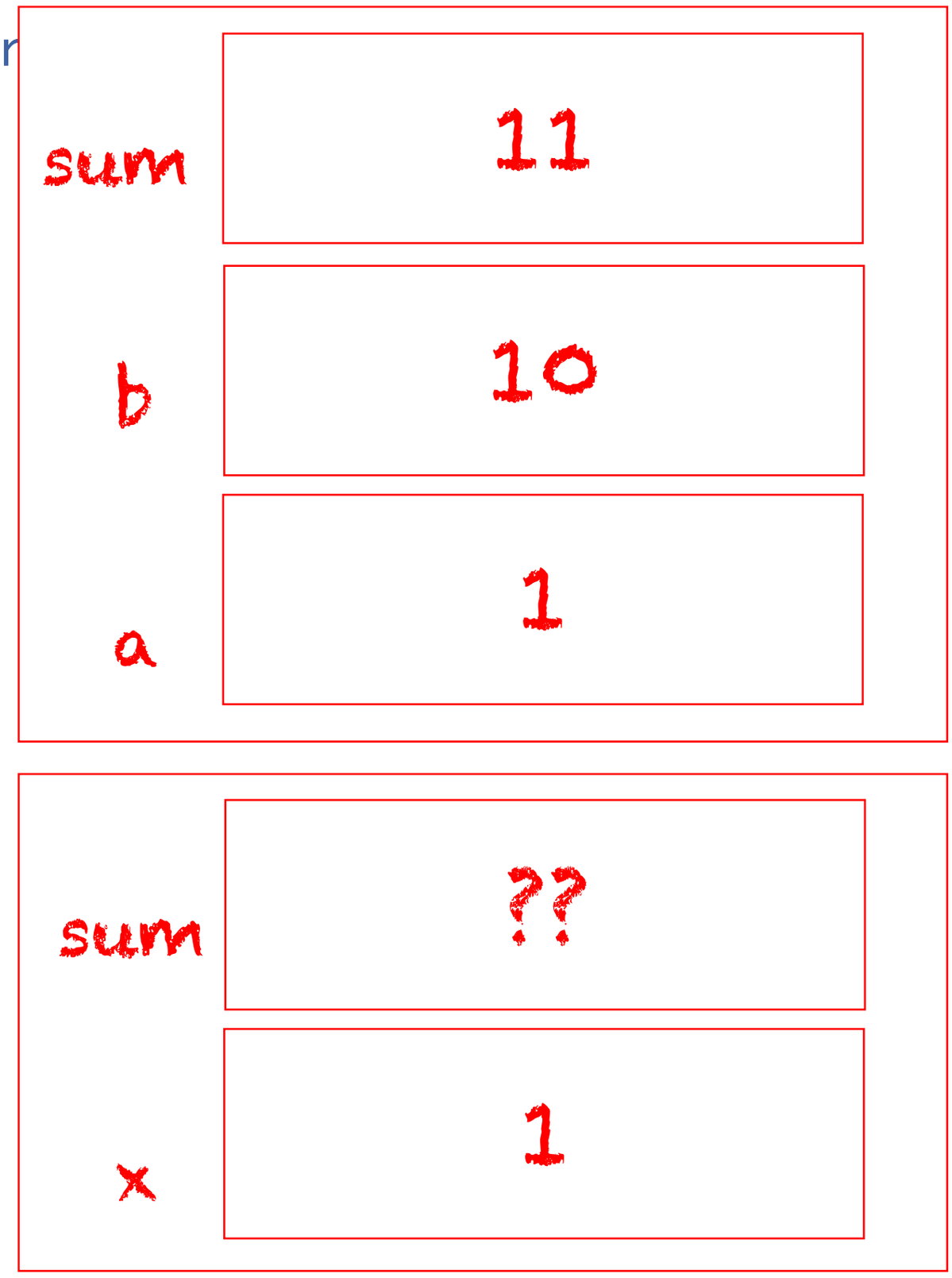
```
void add(long sum, long a, lor
{
   sum = a + b;
}

int main()
{
   long x = 1;
   long sum;
   add(sum, x, 10);
}
```



```
void add(long sum, long a, lor
{
   sum = a + b;
}

int main()
{
   long x = 1;
   long sum;
   add(sum, x, 10);
}
```



```
void add(long sum, long a, long b)
{
  sum = a + b;
}

int main()
{
  long x = 1;
  long sum;
  add(sum, x, 10);
}
```

& address of a variable (i.e, address of a memory location)

* memory location at an address (i.e., the variable at the address)

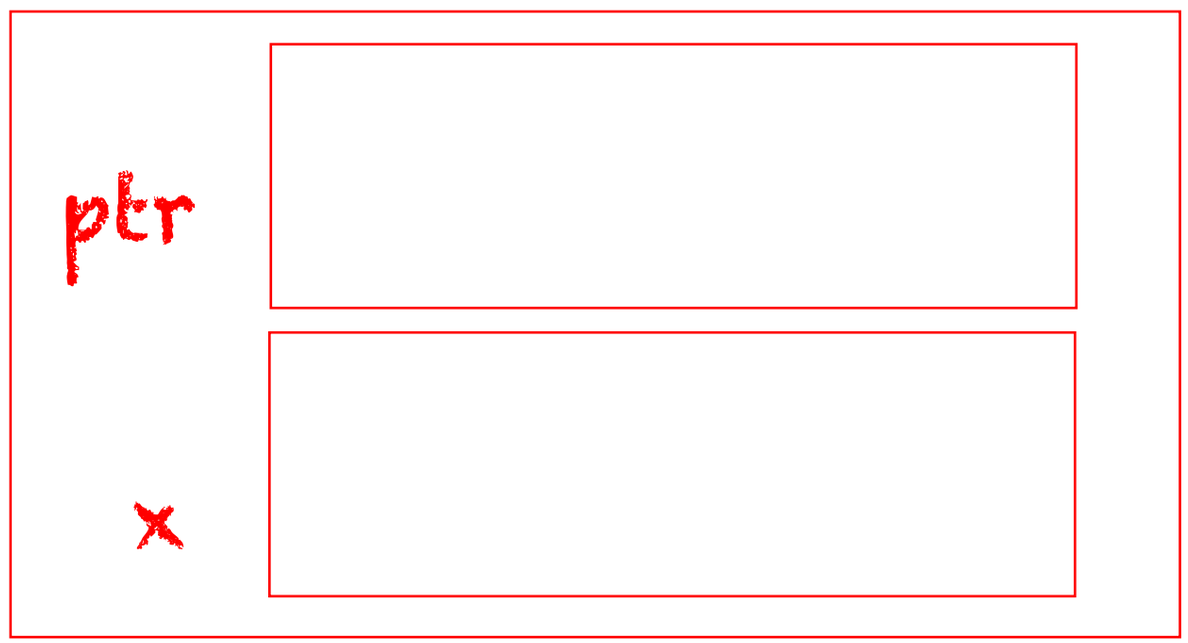
& address of a variable

if x is a variable, then &x gives us the address of x. (where does x live?)

Suppose we have an address a, how to find out who lives there?

answer: *a

```c
int main()
{
  long x;
  long *ptr;
  ptr = &x;
  *ptr = 1;
}
```



```c
#include "cs1010.h"

void add(long sum, long a, long b)
{
  sum = a + b;
  cs1010_println_long((long)&sum);
}

int main()
{
  long x = 1;
  long sum;
  add(sum, x, 10);
  cs1010_println_long((long)&sum);
}
```

Some Rules about Pointers

- A pointer to some type T can only points to a variable of type T
- We cannot change the address of a variable (but can change what a pointer is pointing to)
- We can perform add and subtract on pointers

Arrays

array decay

```
long a[10];
```

a

is equivalent to

```
&a[0]
```

```
long a[2] = {0, 1};
long b[2] = {0, 1};

if (a == b) {  // always false
    :
}

b = a; // not possible
```

```
a[i] = 10;
*(a + i) = 10;
```

```
x = a[42];
x = *(a + 42);
```

```
long max(long list[], long length)
{
   :
}
```

```
int main()
{
   long a[10] = { … };
   max(a, 10);
}
```

```
long max(long *list, long length)
{
   :
}
```

```
int main()
{
   long a[10] = { … };
   max(a, 10);
}
```

```
long a[ ? ];
```

```
long size = cs1010_read_long();
    :
long a[size];
```

variable-size array

```
long size = cs1010_read_long();
   :
long *a = cs1010_read_long_array(size);
```

variable-size array

Strings

A string is an array of char terminated by ‘\0’

```
char *str = “hello!”;
```

```
char str[7] = { ‘h’, ‘e’, ‘l’,
      ‘l’, ‘o’, ‘!’, ’\0’}
```