Laboratory 10 – Pointers – part1

08.12.2020

1. [5points] Please follow the instructions step by step:

a) Declare a variable of type **`float`** `x` and two pointers to **`float`** `pa`, `pb`.
b) Assign a value to the variable `x`.
c) Assign the address of the variable `x` to the pointer `pa`.
d) Assign a null pointer constant to the `pb` pointer. `NULL` is called a null pointer constant. `NULL` is defined in e.g. <`stdio.h`>. Just use it!
e) Print the addresses of all variables.
f) Print the values of all variables.
g) Copy the value of the `pa` pointer into the `pb` pointer.
h) Print the values of all variables.
i) Use the `pb` pointer to modify the value of the variable `x`.
j) Print the values of all variables.
k) Print the values that are at the addresses that the pointers store.

Test data:

The addresses will differ on different computers!

```
e) A: &x = 0x7ffcecd0dd4c,
     &pa = 0x7ffcecd0dd40,
     &pb = 0x7ffcecd0dd38

f) V1: x = 12.51,
      pa = 0x7ffcecd0dd4c,
      pb = (nil)    <-- The printf function prints a null pointer constant as (nil).

h) V2: x = 12.51,
      pa = 0x7ffcecd0dd4c,
      pb = 0x7ffcecd0dd4c

j) V3: x = 3.14,
       pa = 0x7ffcecd0dd4c,
       pb = 0x7ffcecd0dd4c

k) V4: *pa = 3.14,
       *pb = 3.14
```

2. [5points] Please follow the instructions step by step:

a) Declare two variables of type **`double`** `x, y`.
b) Declare two pointers to type **`double`** `p1, p2`. Set them to `NULL` when declaring.
c) Print the addresses of all variables.
d) Print the values of the pointers.
e) Assign the address of the variable `a` to pointer `p1`.

f) Assign the address of the variable `b` to pointer `p2`.
g) Print the values of the pointers.
h) Use the `scanf` function and the pointers `p1` and `p2` to set values to `a` and `b`.

Remember!
The `scan` function uses addresses that are the values of `p1` and `p2`. The function cannot modify the values of `p1` and `p2`, but it can use them to modify `a` and `b`.
You can also pass the addresses of `a` and `b` to the `scanf` function directly without using `p1` and `p2`, but this is not the subject of this exercise.

i) Print the values of all variables.
j) Declare 4 variables of **`double`** type: `sum, diff, prod, quot`. Using the pointers `p1` and `p2` and not using the variables `a` and `b`, calculate the sum, difference, product and quotient of `a` and `b`.
k) Print the results using the variables: `sum, diff, prod, quot, p1, p2`.
Don't use variables `a` and `b`.

Test data:

The addresses will differ on different computers!

```
c)A:  &a = 0x7ffec1871798,
      &b = 0x7ffec1871790,
     &p1 = 0x7ffec1871788,
     &p2 = 0x7ffec1871780

d) V1: p1 = (nil),
       p2 = (nil)

g) V2: p1 = 0x7ffec1871798,
       p2 = 0x7ffec1871790

h) Enter any two real numbers: 12.3 13.4

i) V3: p1 = 0x7ffec1871798,
       p2 = 0x7ffec1871790,
        a = 12.300000,
        b = 13.400000

k) 12.30 + 13.40 = 25.70
   12.30 - 13.40 = -1.10
   12.30 * 13.40 = 164.82
12.30 / 13.40 = 0.92
```

3. [5points] Conversion between coordinate systems.
In mathematics, the polar coordinate system is a two-dimensional coordinate system in which each point on a plane is determined by a distance from a reference point and an angle from a reference direction.

The **Cartesian** coordinates **x** and **y** can be converted to **polar** coordinates **r** and **φ** with $r \geq 0$ and φ in the interval $(-\pi, \pi]$ by:

$r = \sqrt{x^2 + y^2}$

$\varphi = atan2(y, x)$, where **atan2** returns the principal value of the arc tangent of `y/x`, expressed in **radians**.

The **polar** coordinates **r** and **φ** can be converted to the **Cartesian** coordinates **x** and **y** by using the trigonometric functions sine and cosine: $\begin{matrix} x = r\cos\varphi \\ y = r\sin\varphi \end{matrix}$, where **sin** returns the sine of an angle of **φ** radians, **cos** returns the cosine of an angle of **φ** radians.

One radian is equivalent to `180/PI` degrees.

To convert degrees to radians or radians to degrees use the constant PI.

```
#define PI (4.0*atan(1))
```

In the main function:
a) Declare two variables of the type `float x` and `y`.
b) Use the `scanf` function to make `x` and `y` variable values.
c) Declare two variables of the type `float r` and `t`.
d) Call the `CartToPolar` function to which you pass the values of the `x` and `y` variables and the addresses of the `r` and `t` variables.
e) Print the values of the polar coordinates `r` and `t`.
f) Declare two variables of the type `float x1` and `y1`.
g) Call the PolarToCart function to which you will pass the addresses of the variables `x1` and `y1` and the values of the variables `r` and `t`.
h) Print the Cartesian coordinates `x1` and `y1`.

The `CartToPolar` and `PolarToCart` functions take **4** parameters: two `float` type variables, two `float pointer`s. Functions do not use `return` statements and are of type `void`. Use the appropriate formulas to make conversions between the coordinate systems.

Test data:

```
Enter cartesian coordinate x: 1
Enter cartesian coordinate y: 1

Polar coordinate:
rho = 1.41
theta = 45.00 in degree

Cartesian coordinate:
x = 1.00
y = 1.00

Enter cartesian coordinate x: 1
Enter cartesian coordinate y: 8
```

```
Polar coordinate:
rho = 8.06
theta = 82.87 in degree

Cartesian coordinate:
x = 8.00
y = 1.00
```

Next lab 11 – Pointers and arrays and functions.