

COSC2P05 Concepts of Programming Languages

Homework 5

Due: 16:00 pm EST, April 8, 2022 on Sakai

1. In the C program below:

```
void main() {
    int element = 0, arr[6] = {1,4,6,2,0,3};

    sub1(element,arr[0]);
    printf("%d and [%d,%d,%d,%d,%d]\n",
           element,arr[0],arr[1],arr[2], arr[3],arr[4]);

    sub1(arr[0],arr[2]);
    printf("%d and [%d,%d,%d,%d,%d]\n",
           element,arr[0],arr[1],arr[2], arr[3], arr[4]);

    sub1(element,arr[element]);
    printf("%d and [%d,%d,%d,%d,%d]\n",
           element,arr[0],arr[1],arr[2],arr[3],arr[4]);

    return 0;
}

void sub1(int a, int b) {
    int temp;
    temp = a;
    a = b;
    b = temp;

    b=a+b;
}
```

What is the output of the program for each of the parameter-passing methods below:

- a) Passed by value
- b) Passed by reference
- d) Passed by value-result

2. In the program segment in C below, `address()` prints the address of the function `fun`.

```
void fun () {
    printf("\n Hello world!");
}
void address() {
    printf("address: %u", fun);
}
```

a) Re-write `address()` to take in any function and print its address (i.e., parametrize it). Hint, in C you can pass functions as parameters using pointers. Save this program containing the re-written function `address()` as `findAddress.c`.

c) Re-write the Javascript program on the right to C. Run `sub1` using a call in a main program; what is the result? (you can change `alert()` to `printf()`). Based on this result, discuss whether or not C uses deep binding, shallow binding, or ad-hoc binding for reference environments of subprograms passed as parameters.

Save your main program as `test.c`

```
function sub1() {
    var x;
    function sub2() {
        alert(x); // Creates a dialog box with the value of x
    };
    function sub3() {
        var x;
        x = 3;
        sub4(sub2);
    };
    function sub4(subx) {
        var x;
        x = 4;
        subx();
    };
    x = 1;
    sub3();
};
```

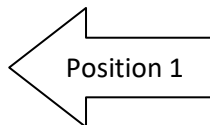
3. For the skeletal JavaScript program in the next page (remember that the choice of programming language I make is often arbitrary and doesn't affect the solution) – perform the following:

- What is the call sequence for the program assuming `bigsub()` is run from some main program, `main()`? This question is a lovely example that we came up with in class together of how recursion – in this case it's indirect recursion – can be performed using parametrization. Note: `Boolean()` is a function in Java that outputs `false` if its parameter is 0, and `true` otherwise.
- Show how the stack looks like for all the subprograms given the call sequence for a), which will include all activation record instances, including static and dynamic chains, when execution reaches position 1. Assume the subprogram `bigsub` has `nest_depth` of 0. Also assume the order of declaration and stack pushes of variables is left-to-right (e.g., in `var y, w`; `y` is pushed first then `w` is pushed next onto the stack).
- For each variable in the expression at position 1, show their chain offset and local offset.
- Show the table for reference variables at position 1 assuming dynamic scope and a shallow access implementation. (this is like the table I showed in the second last slide of Week 10).

```

function bigsub() {
    var y, w;
    function sub1(p1){
        var v;
        function sub2(){
            var u,z;
            ...
            sub1(0);
        } //end of sub2
        if Boolean(p1){
            sub2();
        } else {
            sub3();
        }
        ...
    } //end of sub1
    function sub3(){
        var u,w,y;
        function sub4(){
            var t;
            ...
            z = y*u + t/w;
            ...
        } // end of sub4
        ...
        sub4();
    } // end of of sub3
    var t,v;
    ...
    sub1(1);
    ...
} //end of bigsub

```



What to submit on Sakai:

1. A single pdf (e.g., assignment5.pdf) of your answers to all of the questions above (ideally typed, but you can also paste scans of handwritten answers on the pdf).
2. Your `findAddress.c` file and `test.c` file.
3. Zip these together.