

EE 2905

Dr. Johnson

Homework 5

1 – Identify the error in each function declaration.

10pts

void ave(char s, char t);

none

void wed(sat int, sun int);

type and variable backwards

foo2(char tire, float steer);

no return type

int fun1(int s, t);

no type for t

int foo(float black, char white)

no ;

2 – Identify the error in each function call.

10pts

fun2((char) a, b);

none

ave(float red, blue);

no type in the actual parameters

fri(mon , tue)

no ;

int foo(a, b);

no type in a call

tire(tires steer);

no ,

3 – Identify the error in each function definition.

10pts

```
int foo(float x, int y){  
    x = x + y;  
    return x;  
}
```

OK - wrong type for return  
but it gets converted

```
void foo(int x){  
    int y;  
    ...  
    return y;  
}
```

fn is void – cannot return int

```
int foo(int x, int y){  
    z = x + y;  
    return z;  
}
```

z never declared

4 – Given the following program. What will be printed out for the answer if the user enters 41 as the input? 10pts

```
/*
 * hw6_1.c
 */
#include <stdio.h>

int fun1(int a);
int fun2(int a);
int fun3(int a);

int main(void){
    int a;
    int b;

    printf("Enter an integer: ");
    scanf("%d", &a);

    b = fun1(a);
    printf("answer is: %d",b);

    return 0;
}

int fun1(int b){
    int c;
    c = fun2(b) + fun3(b);
    return c;
}
```

```
int fun2(int d){
    return(d % 10);
}
```

```
int fun3(int e){
    int f;
    f = e/10;
    f = f % 10;
    return f;
}
```

main calls fun1(41)  
fun1 calls fun2(41) and fun3(41)  
fun2(41) returns  $41 \% 10 = 1$   
fun3(41) returns  $4.1 \rightarrow 4 \% 10 = 4$   
fun1 adds the two and returns 5  
main prints 5

5

5 – Given the following memory map – evaluate each item

20pts

```
int foo;  
int boo;  
int zoo;  
int* a_ptr;  
int* b_ptr;
```

variable name	value	address
foo	0x1234	0x1000
boo	0x8000	0x2000
zoo	0x2324	0x3000
a_ptr	0x2000	0x7000
b_ptr	0x3456	0x8000

&boo                    addr of -> 2000

\*b\_ptr                    value pointed to by b\_ptr (value at 0x3456)

a\_ptr                    value of a\_ptr

&b\_ptr                    addr of b\_ptr

boo + b\_ptr + \*a\_ptr + &boo

$$8000 + 3456 + 8000 + 2000$$

HEX

0x2000
?????
0x2000
0x8000
0x15456

## 6 – Fill in the memory map at the end of the following code

20pts

\*\*\* note: this code will not compile – for illustrative purposes only \*\*\*

```
int foo, boo;
float zoo, soo;
int* a_ptr, b_ptr;
float* c_ptr, d_ptr;
```

```
a_ptr = &boo;
c_ptr = a_ptr + 0x2000;
*c_ptr = 3.5 * foo;
*a_ptr = 7;
zoo = soo + boo;
d_ptr = a_ptr + c_ptr;
*d_ptr = 0x1000;
*b_ptr = boo + *a_ptr;
```

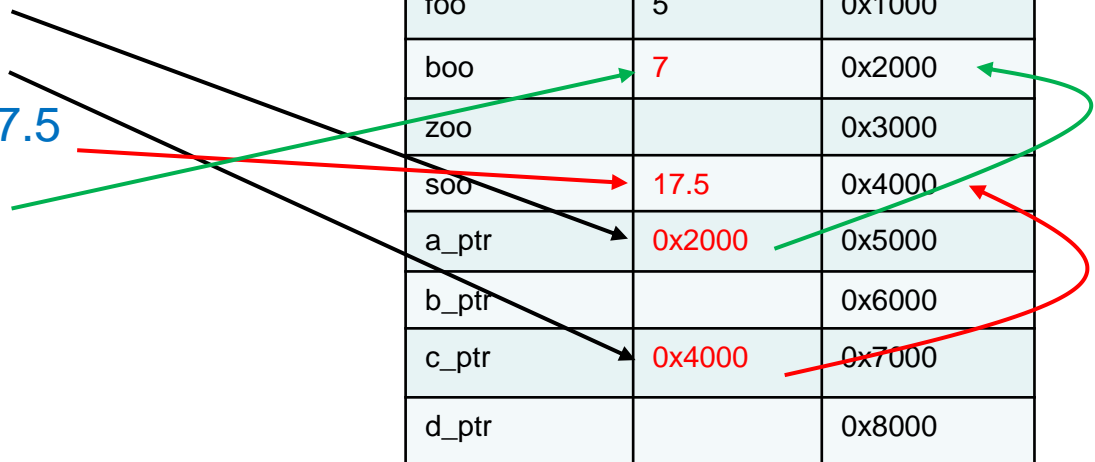
0x2000

0x4000

soo = 17.5

boo = 7

variable name	value	address
foo	5	0x1000
boo	7	0x2000
zoo		0x3000
soo	17.5	0x4000
a_ptr	0x2000	0x5000
b_ptr		0x6000
c_ptr	0x4000	0x7000
d_ptr		0x8000



## 6 – Fill in the memory map at the end of the following code

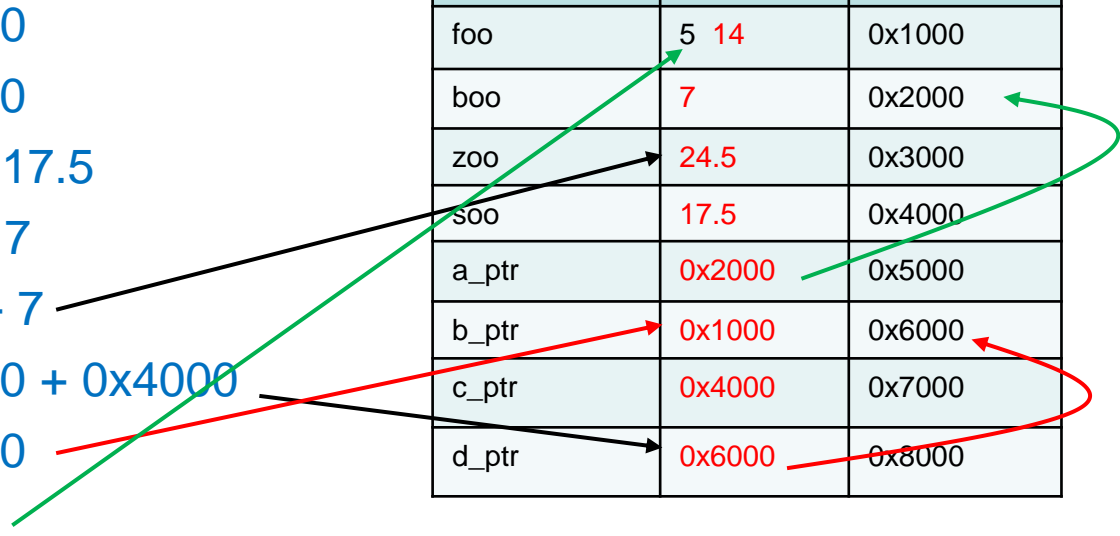
20pts

\*\*\* note: this code will not compile – for illustrative purposes only \*\*\*

```
int foo, boo;
float zoo, soo;
int* a_ptr, b_ptr;
float* c_ptr, d_ptr;
```

```
a_ptr = &boo;           0x2000
c_ptr = a_ptr + 0x2000; 0x4000
*c_ptr = 3.5 * foo;     soo = 17.5
*a_ptr = 7;             boo = 7
zoo = soo + boo;       17.5 + 7
d_ptr = a_ptr + c_ptr;  0x2000 + 0x4000
*d_ptr = 0x1000;       0x1000
*b_ptr = boo + *a_ptr;  7 + 7
```

variable name	value	address
foo	5 14	0x1000
boo	7	0x2000
zoo	24.5	0x3000
soo	17.5	0x4000
a_ptr	0x2000	0x5000
b_ptr	0x1000	0x6000
c_ptr	0x4000	0x7000
d_ptr	0x6000	0x8000





7 – Provide the final values after executing the following code snippet 20pts

```

#include <stdio.h>

int fun1(int* a, int* b);
float fun2(float a, float* b);

int main(void){

    int a = 3;
    int b = 2;
    float c = 3.5;
    float d = 4.5;
    float e = 12;

    a = fun1(&a,&b);
    e = fun2(c,&d);

    return 0;
}

float fun2(float zoo, float* soo){
    zoo = 34;
    *soo *= 3*zoo;
    return *soo;
}

int fun1(int* foo, int* boo ){
    int zoo;
    if (*boo > 0){
        zoo = 2**foo;
    }
    else{
        zoo = 3**foo;
    }
    *boo = zoo + *foo;
    *foo = 12;
    return zoo;
}

```

a 3 -> 12 -> 6  
 b 2 -> 9  
 c 3.5  
 d 3.5 -> 459.0  
 e 12 -> 459.0

a	6
b	9
c	3.5
d	459.0
e	459.0