CSC209H Worksheet: Function Calls and Pointers

1. Trace the memory usage for the program below up to the point when `lie` returns. We have set up both stack frames for you.

```
#include <stdio.h>

void lie(int *age) {
    printf("You are %d years old\n",*age);
   *age += 1;
    printf("You are %d years old\n",*age);
}

int main() {
    int age = 18;
    lie(&age);
    printf("But your age is still %d\n", age);
    return 0;
}
```

Section	Address	Value	Label
stack frame for lie	0x23c	~~18~~ 19	age
	0x240		
	0x244		
	0x248		
	0x24c		
stack frame for main	0x250	18	age
	0x254		
	0x258		
	0x25c		
	0x260		
	0x264		

2. In the space below, modify the above program so that `lie` takes in a pointer so that the change it makes persists after it returns. Trace through your new program (you'll need to write sections and labels yourself).

solution

Section	Address	Value	Label
stack frame for lie	0x23c	0x254	age
	0x240		
	0x244		
	0x248		
	0x24c		
	0x250		
stack frame for main	0x254	~~18~~ 19	age
	0x258		
	0x25c		
	0x260		
	0x264		

CSC209H Worksheet: Function Calls and Pointers

3. In the space below, write a small program that allocates an array of integers in the main function and passes that array to a function call `change`. (You'll also need to pass in the length of the array – **why**?) The function should do two things:

   - Add 10 to each element of the array.
   - Return the average of the new contents of the array.

   Check your understanding carefully by tracing the execution of the function on the given memory model diagram.

```
#include <stdio.h>

float change(int *b, int size){
    int sum =0;
    int i;
    for(i=0, i<size; i++){
        b[i] += 10;
        sum += b[i];
    }
    return (float) sum/ size;
}
int main () {
    int a[4] = {10,20,30,40};
    float result = change(a,4);
    return 0;
}
```

Section	Address	Value	Label
stack frame of change	0x23c	0x25c	b
	0x240		
	0x244	4	size
	0x248	~~0~~ ~~20~~ ~~50~~ ~~90~~ 140	sum
	0x24c	~~0~~ ~~1~~ ~~2~~ ~~3~~ 4	i
	0x250		
	0x254		
	0x258		
stack frame of main	0x25c	~~10~~ 20	a
	0x260	~~20~~ 30	
	0x264	~~30~~ 40	
	0x268	~~40~~ 50	
	0x26c	35.0	result