CSC209H Worksheet: malloc Basics

1. Each time a variable is declared or memory is otherwise allocated, it is important to understand how much memory is allocated, where it will be allocated and when it will be de-allocated. Complete the table below. (Note: some of the programs allocate more than one block of memory.)

Code Fragment	Space?	Where?	De-allocated when?

`int main() {`
`  int i;`
`}`
	sizeof(int)	stack frame for `main`	when program ends

`int fun() {`
`  float i;`
`}`
`int main() {`
`  fun();`
`}`
	sizeof(float)	stack frame of fun	when fun returns

`int fun(char i) {`
`  ...`
`}`
`int main() {`
`  fun('a');`
`}`
	sizeof(char)	stack frame of fun	when fun returns

`int main() {`
`  char i[10] = {'h','i'};`
`}`
	sizeof(char) * 10	stack frame of main	when prog ends

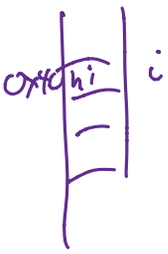
`int main() {`
`  char *i;`
`}`
	sizeof(char*)	4	4

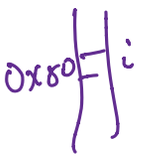
`int main() {`
`  int *i;`
`}`
	sizeof(int*)	4	4

`int fun(int *i) {`
`  ...`
`}`
`int main() {`
`  int i[5] = {4,5,2,5,1};`
`  fun(i);`
`}`
	→ sizeof(int*)
	→ sizeof(i)
	5 * 4
		sf fun
		sf main
			when fun returns
			prog end

`int main() {`
`  int *i;`
`  i = malloc(sizeof(int));`
`}`
	sizeof(int*)
	sizeof(int)
		sf main
		heap
			- when prog end
			- when freed or prog ends

`void fun(int **i) {`
`  *i = malloc(sizeof(int)*7);`
`}`
`int main() {`
`  int *i;`
`  fun(&i);`
`  free(i);`
`}`
	sizeof(int*)
	→ 7*4 = 28
	sizeof(int*)
	(8)
		- sf fun
		heap
		sf main
			→ free





⇒ int i[5]

i is not a pointer
↳ can be used as pointer to each element of the array

i[0] = 11

i = &i[0]

*i = 12

0

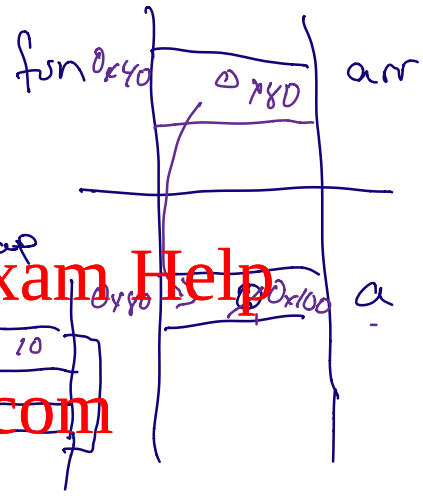
CSC209H Worksheet: malloc Basics

2. Trace the memory usage for the program below up to the point when `initialize` is about to return. We have set up both stack frames for you, and the location of the heap.

```
#include <stdio.h>
#include <stdlib.h>


// Initialize two parallel lists.
void initialize(int *a1, int *a2, int n) {
    for (int i = 0; i < n; i++) {
        a1[i] = i;
        a2[i] = i;
    }
}

int main() {
    int numbers1[3];
    int *numbers2 = malloc(sizeof(int) * 3);

    initialize(numbers1, numbers2, 3);

    for (int i = 0; i < 3; i++) {
        printf("%d %d\n",
               numbers1[i], numbers2[i]);
    }

    free(numbers2);
    return 0;
}
```

Section	Address	Value	Label
Heap	0x23c		
	0x240		
	0x244		
	0x248		
	⋮	⋮	
stack frame for initialize	0x454		
	0x458		
	0x45c		
	0x460		
	0x464		
	0x468		
	0x470		
stack frame for main	0x474		
	0x478		
	0x47c		
	0x480		
	0x484		
	0x488		
	0x48c		

```
void fun(int **arr) {
    *arr = malloc(4*sizeof(int));
    (*arr)[0] = 1;
}

int main() {
    int *a = NULL;
    fun(&a);
    a[0] = 10;
}
```



char dataset[][][];

image →

i