CSCB09H Worksheet: malloc and strings

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();`
`}`			

`int fun(char i) {`
`  ...`
`}`
`int main() {`
`  fun('a');`
`}`			

`int main() {`
`  char i[10] = "hello";`
`}`			

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

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

`int main() {`
`  char *i = "hello";`
 	
`}`			

`int fun(int *i) {`
`  ...`
`}`
`int main() {`
`  int i[5] = {4,5,2,5,1};`
`  fun(i);`
`}`			

`int main() {`
`  int *i;`
`  i = malloc(sizeof(int));`
`}`			

`void fun(int **i) {`
`  *i = malloc(sizeof(int)*7);`
`}`
`int main() {`
`  int *i;`
`  fun(&i);`
`  free(i);`
`}`			

CSCB09H Worksheet: malloc and strings

2. Write a program that declares 3 strings. The first named `first` should be set to the value `"Monday"`, and be stored on the stack frame for `main`. `second` should be a string literal with the value `"Tuesday"`. `third` should have value `"Wednesday"` and be on the heap. The pointers for `second` and `third` will be in stack frame for `main`.

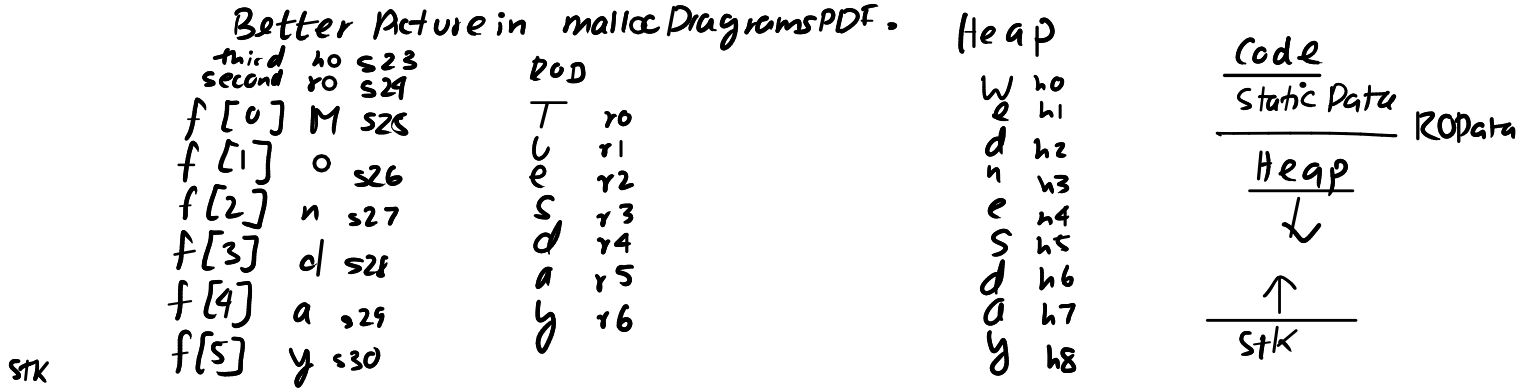
```
char first[6] = "Monday";
char *second = "Tuesday";
char *third = malloc (9*sizeof(char));
strcpy(third, "wednesday");
```

3. Write statements to shorten the strings to the abbreviations for the day names. For example, change `"Monday"` to `"Mon"`. Which string can not be changed in place? Why not?

```
first[3] = '\0'; or strcpy(first, "Mon");
second cannot be changed
third[3] = '\0'; or strcpy(third, "wed");
```

4. Draw the memory model for your program.

Better picture in mallocDiagramsPDF.



5. Add to your program so that it declares an array `string_list` of 3 pointers to char and point the elements to `first`, `second`, and `third`, respectively. So now you have an array of strings. Where is the memory allocated for this array? Add to your picture above.

6. So far much of the allocation has happened in the function `main`. What would happen if you changed `main` to be another function `func` and then returned from it? Which parts of your structure would remain allocated?

5. `char *string_list[3];`
`string_list[0] = first; string_list[1] = second; string_list[2] = third;`
Memory is located in stack. The first and third items are still mutable but the second item is still a string literal in read only Data.

6. Only the dynamic data (third) i.e the data allocated w/ malloc.