

```
1:  /*****
2:   * dma.c
3:   * Doug Lloyd
4:   * October 3, 2010
5:   *
6:   * Fun with dynamic memory allocation
7:   *****/
8:
9:  /* Header files */
10: #include <stdio.h>
11: #include <cs50.h>
12: #include <stdlib.h> // for malloc() and free()
13:
14: /* Function Definitions */
15: int main() {
16:
17:     // Get an integer
18:     printf("Please input an integer: ");
19:     int i = GetInt();
20:
21:     // Make an array that size, statically
22:     int arr1[i];
23:
24:     // Assign its contents, sequentially
25:     for(int a = 0; a < i; a++)
26:         arr1[a] = a;
27:
28:     // Make another array that size, dynamically
29:     int *arr2 = (int *) malloc(sizeof(int) * i);
30:
31:     // Assign its contents, sequentially
32:     for(int a = 0; a < i; a++)
33:         *(arr2 + a) = a;
34:
35:     // Print them both side-by-side
36:     // Notice interoperability of pointers and array syntax
37:     for(int a = 0; a < i; a++) {
38:         printf("arr1[%d] = %d; *(arr2 = %d) = %d\n",
39:             a, *(arr1 + a), a, arr2[a]);
40:     }
41:
42:     // Give memory back!
43:     free(arr2);
44:     return 0;
45: }
46:
47:
```

```
1:  /******
2:  * pointers.c
3:  * Doug Lloyd
4:  * October 3, 2010
5:  *
6:  * Fun with pointers
7:  *****/
8:
9:  /* Header files */
10: #include <stdio.h>
11: #include <cs50.h>
12:
13: /* Function Definitions */
14: int main() {
15:
16:     // Get an integer
17:     printf("Please input an integer: ");
18:     int i = GetInt();
19:
20:     // Show its location
21:     printf("That integer is located at memory address: %X\n", &i);
22:
23:     // Create a pointer, have it point to i
24:     int *pi;
25:     pi = &i;
26:
27:     // Show its contents
28:     printf("The value of pi, the pointer to i, is: %X\n", pi);
29:     printf("Which means that *pi is: %d\n", *pi);
30:
31:     // Get another integer
32:     printf("Change the value of i by how much: ");
33:     int j = GetInt();
34:
35:     // Change it by way of the pointer
36:     *pi += j;
37:
38:     printf("The new value of i is: %d\n", *pi);
39:
40:     return 0;
41: }
42:
43:
```

```
1:  /*****
2:  * struct.c
3:  * Doug Lloyd
4:  * October 3, 2010
5:  *
6:  * Fun with structs
7:  *****/
8:
9:  /* Header files */
10: #include <stdio.h>
11: #include <cs50.h>
12: #include <unistd.h>
13:
14: /* Structure Declarations */
15: struct cat_t {
16:     string name;
17:     int age;
18:     char gender;
19: };
20:
21: /* Function Declarations */
22: struct cat_t makeCat(string n, int a, char g);
23: void printCat(struct cat_t c);
24:
25: /* Function Definitions */
26: int main() {
27:
28:     // Get some info
29:     printf("What is your cat's name? ");
30:     string name = GetString();
31:     char gender;
32:     do {
33:         printf("And is it a male (M) or a female (F)? ");
34:         gender = GetChar();
35:     } while(gender != 'M' && gender != 'F');
36:     string prompt = (gender == 'M') ? "he" : "she";
37:     printf("Lastly, how old is %s? ", prompt);
38:     int age = GetInt();
39:
40:     printf("Thanks. I'll make a record for your cat now\n");
41:     sleep(1);
42:     printf("Making record...\n");
43:     struct cat_t mycat = makeCat(name, age, gender);
44:     sleep(1);
45:     printf("Record complete!\n");
46:     printCat(mycat);
47:     return 0;
48: }
49:
50: struct cat_t makeCat(string n, int a, char g) {
51:     struct cat_t xcat;
52:     xcat.name = n;
53:     xcat.age = a;
54:     xcat.gender = g;
55:     return xcat;
56: }
57:
58: void printCat(struct cat_t c) {
59:     printf("\nName: %s", c.name);
60:     printf("\nAge: %d", c.age);
61:     printf("\nGender: %c\n", c.gender);
62:     return;
63: }
64:
```

```
1:  /*****
2:  * structdma.c
3:  * Doug Lloyd
4:  * October 3, 2010
5:  *
6:  * Fun with dynamically-allocated
7:  * pointers to structs
8:  *****/
9:
10: /* Header files */
11: #include <stdio.h>
12: #include <cs50.h>
13: #include <unistd.h>
14: #include <stdlib.h>
15:
16: /* Structure Declarations */
17: struct cat_t {
18:     string name;
19:     int age;
20:     char gender;
21: };
22:
23: /* Function Declarations */
24: void makeCat(struct cat_t *xcat, string n, int a, char g);
25: void printCat(struct cat_t *c);
26:
27: /* Function Definitions */
28: int main() {
29:
30:     // Get some info
31:     printf("What is your cat's name? ");
32:     string name = GetString();
33:     char gender;
34:     do {
35:         printf("And is it a male (M) or a female (F)? ");
36:         gender = GetChar();
37:     } while(gender != 'M' && gender != 'F');
38:     string prompt = (gender == 'M') ? "he" : "she";
39:     printf("Lastly, how old is %s? ", prompt);
40:     int age = GetInt();
41:
42:     printf("Thanks. I'll make a record for your cat now\n");
43:     sleep(1);
44:     printf("Making record...\n");
45:     struct cat_t *mycat = malloc(sizeof(struct cat_t));
46:     makeCat(mycat, name, age, gender);
47:     sleep(1);
48:     printf("Record complete!\n");
49:     printCat(mycat);
50:     free(mycat);
51:     return 0;
52: }
53:
54: void makeCat(struct cat_t *xcat, string n, int a, char g) {
55:     xcat->name = n;
56:     xcat->age = a;
57:     xcat->gender = g;
58:     return;
59: }
60:
61: void printCat(struct cat_t *c) {
62:     printf("\nName: %s", c->name);
63:     printf("\nAge: %d", c->age);
64:     printf("\nGender: %c\n", c->gender);
```

```
65:    return;  
66: }  
67:
```

```
1:  /*****
2:  * typedef.c
3:  * Doug Lloyd
4:  * October 3, 2010
5:  *
6:  * Fun with typedef
7:  *****/
8:
9:  /* Header files */
10: #include <stdio.h>
11: #include <cs50.h>
12: #include <unistd.h>
13:
14: /* Structure Declarations */
15: typedef struct _cat_t {
16:     string name;
17:     int age;
18:     char gender;
19: } cat_t;
20:
21: /* Function Declarations */
22: cat_t makeCat(string n, int a, char g);
23: void printCat(cat_t c);
24:
25: /* Function Definitions */
26: int main() {
27:
28:     // Get some info
29:     printf("What is your cat's name? ");
30:     string name = GetString();
31:     char gender;
32:     do {
33:         printf("And is it a male (M) or a female (F)? ");
34:         gender = GetChar();
35:     } while(gender != 'M' && gender != 'F');
36:     string prompt = (gender == 'M') ? "he" : "she";
37:     printf("Lastly, how old is %s? ", prompt);
38:     int age = GetInt();
39:
40:     printf("Thanks. I'll make a record for your cat now\n");
41:     sleep(1);
42:     printf("Making record...\n");
43:     cat_t mycat = makeCat(name, age, gender);
44:     sleep(1);
45:     printf("Record complete!\n");
46:     printCat(mycat);
47:     return 0;
48: }
49:
50: cat_t makeCat(string n, int a, char g) {
51:     cat_t xcat;
52:     xcat.name = n;
53:     xcat.age = a;
54:     xcat.gender = g;
55:     return xcat;
56: }
57:
58: void printCat(cat_t c) {
59:     printf("\nName: %s", c.name);
60:     printf("\nAge: %d", c.age);
61:     printf("\nGender: %c\n", c.gender);
62:     return;
63: }
64:
```