list0.c

```c
// Implements a list of numbers using an array of fixed length

#include <cs50.h>
#include <stdio.h>

int main(void)
{
    // Prompt for number of numbers
    int capacity;
    do
    {
        capacity = get_int("Capacity: ");
    }
    while (capacity < 1);

    // Memory for numbers
    int numbers[capacity];

    // Prompt for numbers
    int size = 0;
    while (size < capacity)
    {
        // Prompt for number
        int number = get_int("Number: ");

        // Add to list
        numbers[size] = number;
        size++;
    }

    // Print numbers
    for (int i = 0; i < size; i++)
    {
        printf("%i\n", numbers[i]);
    }
}
```

```c
// Implements a list of numbers using an array of dynamic length

#include <cs50.h>
#include <stdio.h>

int main(void)
{
    // Memory for numbers
    int *numbers = NULL;
    int capacity = 0;

    // Prompt for numbers (until EOF)
    int size = 0;
    while (true)
    {
        // Prompt for number
        int number = get_int("Number: ");

        // Check for EOF
        if (number == INT_MAX)
        {
            break;
        }

        // Check whether enough space for number
        if (size == capacity)
        {
            // Allocate space for number
            int *tmp = realloc(numbers, sizeof(int) * (size + 1));
            if (!tmp)
            {
                if (numbers)
                {
                    free(numbers);
                }
                return 1;
            }
            numbers = tmp;
            capacity++;
        }

        // Add number to list
        numbers[size] = number;
        size++;
    }

    // Print numbers
    printf("\n");
    for (int i = 0; i < size; i++)
    {
        printf("%i\n", numbers[i]);
    }

    // Free memory
    if (numbers)
    {
        free(numbers);
    }
}
```

```c
// Implements a list of numbers using a linked list

#include <cs50.h>
#include <stdio.h>

typedef struct node
{
    int number;
    struct node *next;
}
node;

int main(void)
{
    // Memory for numbers
    node *numbers = NULL;

    // Prompt for numbers (until EOF)
    while (true)
    {
        // Prompt for number
        int number = get_int("number: ");

        // Check for EOF
        if (number == INT_MAX)
        {
            break;
        }

        // Allocate space for number
        node *n = malloc(sizeof(node));
        if (!n)
        {
            return 1;
        }

        // Add number to list
        n->number = number;
        n->next = NULL;
        if (numbers)
        {
            for (node *ptr = numbers; ptr != NULL; ptr = ptr->next)
            {
                if (!ptr->next)
```

```c
                    {
                        ptr->next = n;
                        break;
                    }
                }
            }
            else
            {
                numbers = n;
            }
        }

        // Print numbers
        printf("\n");
        for (node *ptr = numbers; ptr != NULL; ptr = ptr->next)
        {
            printf("%i\n", ptr->number);
        }

        // Free memory
        node *ptr = numbers;
        while (ptr != NULL)
        {
            node *next = ptr->next;
            free(ptr);
            ptr = next;
        }
    }
```

```c
// http://valgrind.org/docs/manual/quick-start.html#quick-start.prepare

#include <stdlib.h>

void f(void)
{
    int *x = malloc(10 * sizeof(int));
    x[10] = 0;
}

int main(void)
{
    f();
    return 0;
}
```

struct0.c

```c
// Demonstrates structs

#include <cs50.h>
#include <stdio.h>
#include <string.h>

#include "struct.h"

int main(void)
{
    // Allocate space for students
    int enrollment = get_int("Enrollment: ");
    student students[enrollment];

    // Prompt for students' names and dorms
    for (int i = 0; i < enrollment; i++)
    {
        students[i].name = get_string("Name: ");
        students[i].dorm = get_string("Dorm: ");
    }

    // Print students' names and dorms
    for (int i = 0; i < enrollment; i++)
    {
        printf("%s is in %s.\n", students[i].name, students[i].dorm);
    }
}
```

struct1.c

```c
// Demonstrates file I/O

#include <cs50.h>
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

#include "struct.h"

int main(void)
{
    // Allocate memory for students
    int enrollment = get_int("Enrollment: ");
    student students[enrollment];

    // Prompt for students' names and dorms
    for (int i = 0; i < enrollment; i++)
    {
        students[i].name = get_string("Name: ");
        students[i].dorm = get_string("Dorm: ");
    }

    // Save students to disk
    FILE *file = fopen("students.csv", "w");
    if (file)
    {
        for (int i = 0; i < enrollment; i++)
        {
            fprintf(file, "%s,%s\n", students[i].name, students[i].dorm);
        }
        fclose(file);
    }
}
```

```c
// Represents a student

typedef struct
{
    char *name;
    char *dorm;
}
student;
```