

Programación Orientada a Datos - ANSI C

Organización del Computador II

Primer Cuatrimestre 2022

En este taller vamos a trabajar con código C interpretándolo desde la perspectiva de los datos, y en particular, de la forma en que los datos se ubican en memoria. También vamos a ejercitar el uso de una pila implementada en C. Los ejercicios correspondientes al primer checkpoint deben completarlos a partir del análisis del código que descargaron para el taller, mientras que el checkpoint 2 se concentra en la práctica de programación donde deberán completar el código para obtener un resultado correcto de la aplicación.

Al corregir cada **checkpoint** todos los miembros del grupo deben estar presentes, salvo aquellas instancias en que puedan justificar su ausencia previamente. Si al finalizar la práctica de la materia algún miembro cuenta con mas de un 20 % de ausencia no justificada en la totalidad de los checkpoints se considerará que cursada como desaprobada.

Cada checkpoint se presenta a lx docente asignadx durante el transcurso de la clase práctica actual o la siguiente, al igual que en la primera actividad, se recomienda subir los archivos completos al git (en esta actividad pueden armar un repo para todo el grupo).

Introducción

En el campus encontraran el archivo `ejC2-bundle.v0.1.tar.gz` conteniendo el presente enunciado junto con el código fuente (incompleto) de un sencillo programa que busca imprimir por consola el contenido de dos tipos de estructuras.

El mismo implementa una pila global que utilizaremos para el movimiento de datos entre funciones. El código fuente de la pila está disponible en `stack.h` y `stack.c` dónde también se implementan algunas funciones útiles.

Los archivos `student.h` y `student.c` definen las estructuras en cuestión y contienen dos funciones `printStudent()` y `printStudentp()` que deberemos implementar. Notar la particularidad que las funciones no reciben parámetros (y no podemos modificar su firma), pero conocen la existencia de la pila global, por ello deberemos idear una forma para poder pasarle a las funciones en cuestión los datos necesarios para que puedan imprimir por consola los datos de cada *estudiante*.

1. Análisis

En este checkpoint vamos a estudiar porciones de código vinculado a la inicialización de arreglos, variables, structs, etc. Ya sea de forma dinámica o estática. También vamos a conceptualizar el funcionamiento de la pila implementada y analizar el comportamiento de ciertas funciones. Finalmente, vamos tratar de construir una idea del código que implementaremos en el checkpoint 2.

1.1. Memoria

- a) ¿En qué segmento de memoria (heap, stack, text, etc.) se encuentra cada variable enumerada a continuación? (Las mismas se encuentran en `main.c`)

Variable	Segmento	Descripción
<code>stack</code>		
<code>*stack</code>		
<code>st1</code>		
<code>stp</code>		
<code>*(stack->pop)</code>		

- b) ¿Que porciones de memoria deberán ser explícitamente liberadas?
- c) Dibujar un diagrama de la organización en memoria de las estructuras `student_t` y `studentp_t` con la alineación de datos correspondiente (ver `student.h`).

1.2. Pila

Mirando los archivos `stack.h`

- a) ¿Qué variable apunta al segmento de memoria utilizado por la pila para almacenar los datos?

- b) ¿Qué ancho tiene la pila (i.e. tamaño del dato que se pushea/popea)?
- c) Dibujar un esquema de la pila, con su organización en memoria y sus miembros.
- d) ¿Qué hace la función `createFrame`? Discutir sobre cuál puede ser el objetivo de la misma.
- e) ¿Qué hace la función `myCall(void (*func)())`? ¿Está asumiendo algo en particular sobre `func()`?

Checkpoint 1

2. Práctica de programación

En este checkpoint vamos a implementar porciones de código vinculado al trabajo con datos y uso de una pila.

Se pide: Completar `main.c` y `student.c` (sin modificar la firma de las funciones) para que:

- a) El programa pueda imprimir correctamente los estudiantes y el valor random pusheado, por ejemplo:

```
Valor: 42
Nombre: Steve Balmer
dni: 12345678
Calificaciones: 3, 2, 1,
Concepto: -2
-----
Nombre: Linus Torvalds
dni: 23456789
Calificaciones: 9, 7, 8,
Concepto: 1
-----
```

- b) Cada función de impresión tenga solamente acceso a su propio frame de la pila (considerando solamente el acceso a través de `push` y `pop`, no la manipulación de punteros)

Además:

- a) Redactar sintéticamente el procedimiento para pasar parámetros a las funciones llamadas por `myCall()`. ¿Con qué hay que ser cuidadosos?
- b) Verificar el correcto manejo de la memoria con la herramienta Valgrind.

Importante: Ver videos introductorios a las herramientas en: <https://campus.exactas.uba.ar/course/view.php?id=3014§ion=9>

Checkpoint 2
