TC1033

Pensamiento Computacional Orientado a Objetos

PROFESOR

- Instructor
  - Ariel Lucien García Gamboa
  - Mentor desde 2019 y profesor de computación desde 2010
- Clase
  - Miércoles y Viernes de 13 a 15 horas
- Medios de Contacto
  - WhatsApp 5547662316
  - ariel.garcia@tec.mx
- Asesorías
  - Escríbanme y nos ponemos de acuerdo

POLÍTICAS DEL CURSO

Durante las sesiones. Es muy importante que durante la sesión mantengas tu micrófono apagado y que sólo lo prendas cuando quieras hacer una aportación o cuando tengas dudas. No necesitas pedir permiso para hablar.

Duración de las sesiones. La clase inicia a las 13:03 en punto (CTM) y estaremos conectados durante 52 minutos. A las 13:55 no detendremos y tomaremos un receso de 5 minutos. A las 14 horas en punto reiniciaremos y finalizaremos la clase a las 14:55 horas. Esto permitirá que tengas un pequeño receso para tomar agua y estirarte

Falta de Integridad Académica. Éste curso, así como todos, prohíbe estrictamente la copia de código (o cualquier material). Pero, ¿eso significa que no puedo colaborar, preguntar, trabajar con mis compañeros? No, significa que tú debes proponer tus propias soluciones.

¿Qué pasa si no entiendo un tema y tengo que entregar una tarea? Fácil, sólo debe explicar ampliamente qué es lo que no entiendes, qué material has revisado por tu cuenta y cuál es la solución que propones (esto te permitirá tener una calificación como si hubieras resuelto el ejercicio/tarea).

POLÍTICAS DEL CURSO

Preguntas. Es muy importante que antes de que expongas una duda revises el material de las sesiones. NO SE VALE PREGUNTAR SIN INVESTIGAR AL RESPECTO EN EL MATERIAL DE LA CLASE. NO SE VALE PREGUNTAR POR LO QUE ACABO DE DECIR (sí, si no te quedó claro lo que dije, entonces expresa que sí entendiste de lo que acabo de decir y luego ya lo que no entendiste). NO SE VALE LLEGAR A LA ASESORÍA Y DECIR "NO ENTENDÍ NADA", "EXPLÍCAME TODO". Es necesario que asuman su responsabilidad de estudiar fuera de clase utilizando el material de las sesiones o investigando en libros o medios digitales.

****Revisa el material que está disponible en Canvas y en github****

Competencias de Área

SICT0300. Solución de problemas con computación

Soluciona problemas de diferentes niveles de complejidad mediante la aplicación de metodologías computacionales y de tecnologías de información en ambientes controlados y de incertidumbre.

Subcompetencias	Nivel de dominio
SICT0302. Toma decisiones en la solución de problemas en condiciones de incertidumbre y diferentes niveles de complejidad con base en metodologías de investigación y de cómputo.	A
SICT0303. Implementa acciones científicas e ingenieriles o procesos computacionales que cumplen con el tipo de solución requerida.	A

SICT0400. Aplicación de estándares internacionales

Aplica estándares, normas y principios de sustentabilidad en el desarrollo de sistemas computacionales y de tecnologías de información.

Subcompetencia	Nivel de dominio
SICT0401. Aplica los estándares y normas en el ejercicio de su profesión manteniéndolos como referencia a seguir en la solución de problemas computacionales y de tecnologías de información.	A

FINES DE APRENDIZAJE

Competencia Transversal

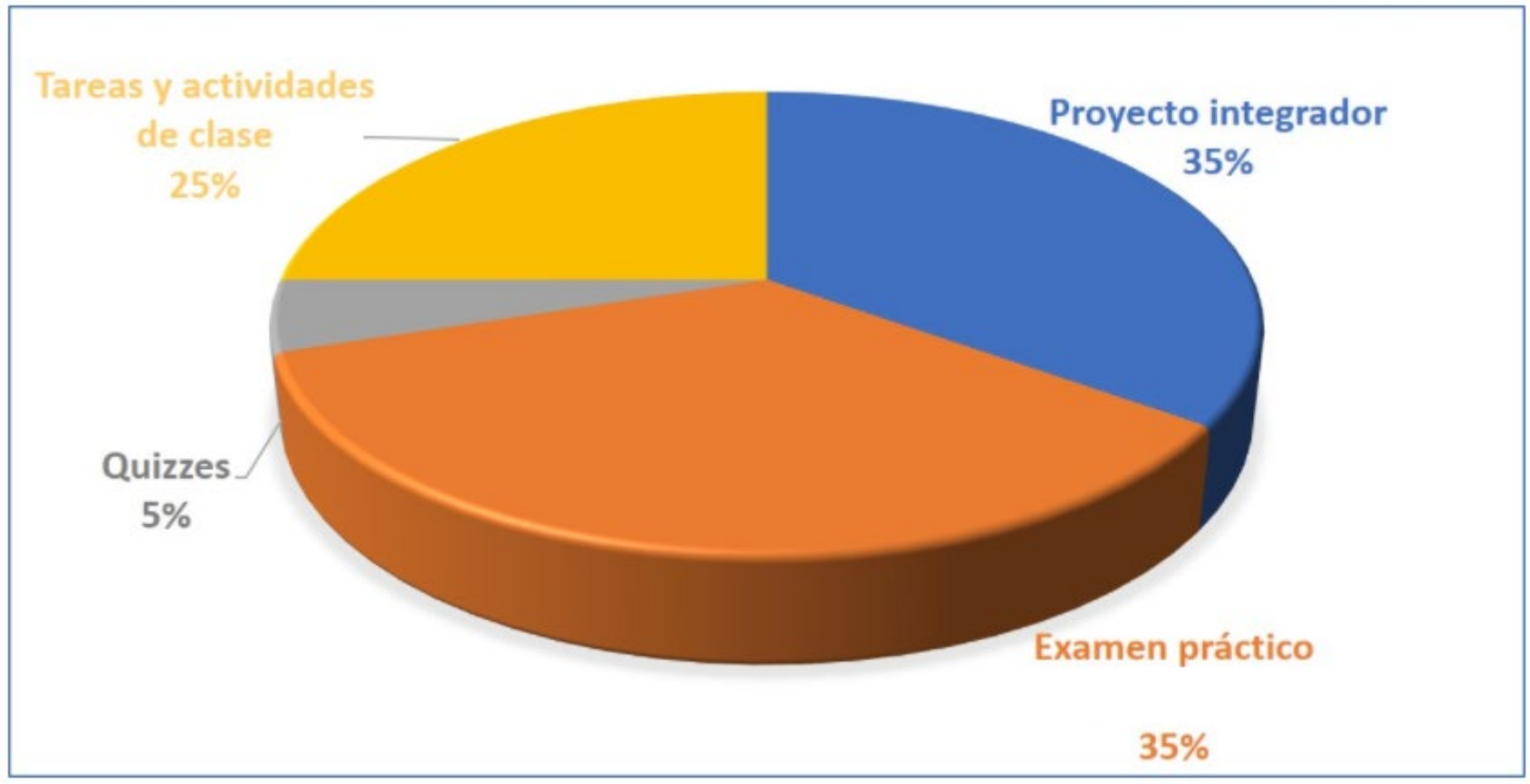
SEG0700. Transformación digital

Genera soluciones a las problemáticas de su ámbito profesional con la incorporación inteligente y oportuna de tecnologías digitales de vanguardia.

Subcompetencia	Nivel de dominio
SEG0702. Tecnologías de vanguardia. Evalúa diversas tecnologías, con apertura a la búsqueda e implementación de alternativas relevantes en la transformación de la práctica profesional.	A

FINES DE APRENDIZAJE

PLAN DE EVALUACIÓN



METODOLOGÍA

Las sesiones y las asesorías serán en zoom. Voy al campus un par de días, así que si requieres una asesoría presencial, escríbeme por whats ☺

Lenguaje: durante el curso aprenderás a programar en C++

IDE de desarrollo: **debes instalar el compilador de C++ y el IDE de desarrollo Visual Studio Code (para la segunda semana ya lo debes tener instalado)**

- En Windows se recomienda instalar Linux Subsytem
- En Mac se recomienda usar Xcode y HomeBrew

Metodología

Explicación de temas
Resolución de ejercicio de programación
Trabajo individual y colaborativo (pares, tercias, etc.)
Participación dentro de la clase

SESIONES

- En Canvas se publicarán las grabaciones de las sesiones
- El curso tiene habilitado un repo de GitHub en el que subiré
  - Liga de la sesión grabada (también estará en Canvas)
  - Presentación de la sesión (también estará en Canvas)
  - Código realizado en clase
- https://github.com/ariellucientec/tc1030_2021113

ARRANQUE DEL CURSO

C++ INTRODUCCIÓN

- Nacido hace casi 40 años sigue siendo uno de los lenguajes de programación más utilizados en el mundo
- Ventajas
  - VELOCIDAD. Se puede crear software muy veloz
  - Permite al programador administrar la memoria (y en consecuencia crear programas muy eficientes)
- Desventajas
  - Las ventajas se pueden volver desventajas para el programador (un gran poder conlleva una gran responsabilidad)

C++ INTRODUCCIÓN

- C++ es un lenguaje compilado (y NO interpretado como es Python)
- El compilador es un programa que traduce código fuente en un programa ejecutable.
- El proceso de traducción tiene 3 etapas: Preprocesamiento, Compilación y Enlazado
- Preprocesamiento. Se revisan la directivas para el compilador
- Compilación. Se traduce el código y se genera código intermedio (object files)
- Enlazado. Se produce el archivo ejecutable final a partir de los object files

Compilador	Interprete
Escanea el código completo, se traduce a Código Máquina y se ejecuta	Se escanea una línea de código, se traduce y se ejecuta
El análisis y procesamiento del código es muy lenta, pero la velocidad de ejecución muy rápida	Y el análisis y procesamiento del código es muy rápido, pero la ejecución es muy lenta
Se genera una lista completa de errores de todo el programa	Se identifican errores únicamente de la línea que se revisa

UN PROGRAMA EN C++

```
#include "arithmetic.h"
#include "wolfram_alpha.h"

int main()
{
    arithmetic::add(5,5);
    wolfram_alpha::add(8,8);
}
```

TIPOS DE DATOS NATIVOS EN C++

Nombre	Descripción	Tamaño en memoria
Int	Números enteros	4 bytes
Float	Punto flotante de precisión simple	4 bytes
Double	Punto flotante de doble precisión	8 bytes
Char	Caracter	1 byte
Bool	Booleano (verdadero / falso)	
Short int	Entero corto	2 bytes
Long int	Entero largo /muy largo	8 bytes

De manera adicional agregaremos como tipo de datos **VOID** que indica Sin Valor (lo usaremos más adelanta con funciones

TIPOS DE DATOS Y VARIABLES

- Los tipos de datos Nativos en C++ permiten declarar variables para que nuestro programa las use
- Las variables permiten almacenar valores que pueden cambiar
- En C++ las variables se declaran utilizando el tipo de datos que almacenarán y colocado un nombre (que no sea una palabra reservada del lenguaje

```
bool repruebo = true;
bool apruebo = false;
```

```
int resultado = 0;
```

```
int consecutivo = 0;
int sum = 0;
```

INDENTACIÓN Y SCOPE Y CIERRE DE INSTRUCCIONES

- Para C++ la Indentación **NO** es relevante. En su lugar se utilizan llaves { } para determinar el *scope* (qué tan grande es un bloque de código)
- Aunque para C++ no es relevante, debes tener tu código perfectamente indentado, de lo contrario NO lo revisaré
  - Esto forma parte de los estándares de programación (recuerda que hay una competencia para evaluar estándares
- El cierre de una instrucción debe realizarse colocando un punto y coma al final de la instrucción de código

UN PROGRAMA EN C++

```cpp
#include<iostream>
#include<stdlib.h>
#include"arithmetic.h"
#include"wolfram_alpha.h"

#define PI = 3.1416;

int resuelveSituacionProblema(int a, double b)
{
    int resultado = 0;
    return resultado;
}

bool noMePongoLasPilasEnElCurso(bool melaspongo)
{
    bool repruebo = true;
    bool apruebo = false;
    if (melaspongo == false)
        return repruebo;
    else
        return apruebo;
}

int main()
{
    resuelveSituacionProblema(10, 8.8);
    noMePongoLasPilasEnElCurso(false);
}
```

Directivas del preprocesador
Librerías

Directivas del preprocesador
Definición de constantes

Funciones

Función principal / Código principal

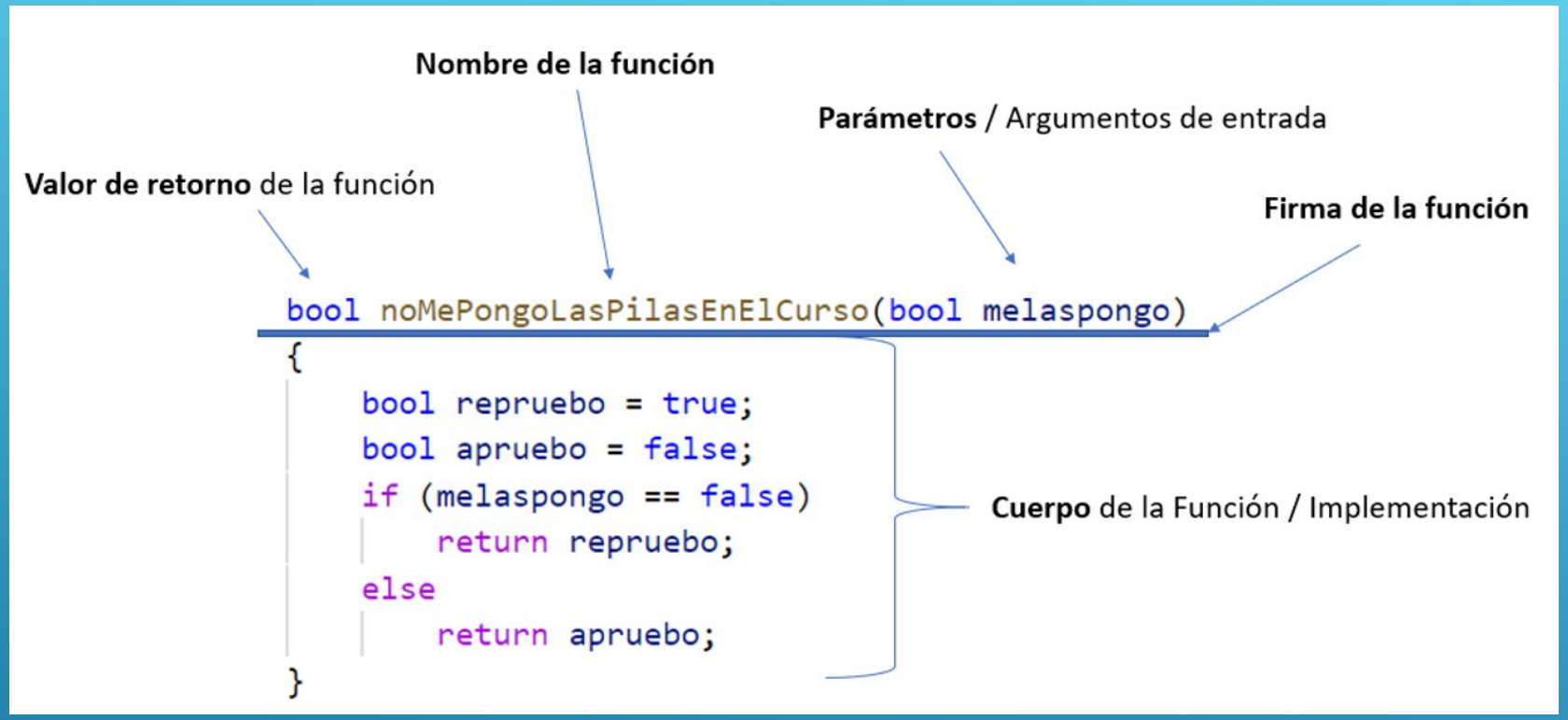
PARTES DE UN PROGRAMA EN C++

- Directivas del preprocesador
  - Permite darle instrucciones al compilador como: agrega una librería (#include), define una constante (#define), entre otras que no usaremos
- Funciones de código
  - Permite que tu código sea modular y que se pueda fácilmente reutilizar. Más adelante aprenderemos otra forma para reutilizar código
- Función Principal
  - Si quieres que tu código realice una serie de acciones, debe haber una y sólo una función principal (main)

FUNCIONES

- Permiten reutilizar código (hacen modular nuestro programa)

- Está compuesta por:
  - Firma de la función
    - Valor de retorno
    - Nombre
    - Parámetros de entrada
  - Cuerpo de la función o Implementación
    - Instrucciones que se van a ejecutar (declaración de variables, operaciones, etc.)
    - Delimitado por llaves
    - Palabra reservada return (para retornar un valor)



FUNCIONES EN C++

MI PRIMER PROGRAMA EN C++

- Entra a la página https://www.onlinegdb.com/online_c++_compiler

- Para la siguiente semana ya debes tener instalado
  - El compilador de C++
  - El IDE de desarrollo Visual Studio Code

- Crea un primer programa

IMPRIMIR EN PANTALLA

- Es necesario agregar una librería con funciones para manipular streams de entrada y salida #include<iostream)
  - Como las funciones están dentro de una librería, entonces es necesario indicar el nombre utilizando el operador de resolución de conflictos :: (scope resolution operator)
- La función para imprimir es cout (console output)
  - Se tiene que utilizar en conjunto con el operador stream insertion <<
- La función para leer valores es cin (console input)
  - Se tiene que utilizar en conjunto con el operador stream extractor >>