**Instituto Superior de Formación Técnica Nº 151 logo151-trans.gif  
Carrera: Analista de Sistemas  
1 Año. Algoritmos y Estructuras de Datos I**

|  |  |
| --- | --- |
| **Trabajo Práctico Nº 3** | **Unidad 3.1** |
| **Modalidad:** Semi-Presencial | **Estratégica Didáctica:** Trabajo Grupal. |
| **Metodología de Desarrollo:** acordar | **Metodología de Corrección:** acordar docente |
| **Carácter de Trabajo:** Obligatorio – con Nota | **Fecha Entrega:** A confirmar por el Docente. |

**Marco Teórico:**

1. Relacionar los conceptos de Tipo, Variable, Valor y Descripción, dar ejemplos.
2. ¿Qué se entiende por Promoción y Conversión de tipos?
3. Explicar y ejemplificar el concepto de Conversión Segura y no segura
4. ¿Qué significa “Cast” o “Castear en castellanización”? dar ejemplos.
5. ¿Para qué casos usaría un tipo Definido por el Usuario? ¿En qué caso lo usaría?
6. ¿Qué es un tipo de Dato enumerado?, ¿en qué casos los utilizaría?
7. Explicar la sentencia que compone la Definición e Inicialización del Tipo enumerado con   
   un ejemplo, describiendo cada una de sus partes.
8. Si bien vimos tipos Enumerados sin ámbito (luego veremos con Ámbito), los tipos Enumerados pueden “Castearse”… buscar un ejemplo y comentarlo.
9. Las “Constantes Simbólicas” se enumeran desde….. (qué valor)
10. Explicar la siguiente “Casting Implícito” -> int num=tortuga dentro de un programa que use enumeradores.

**Marco Práctico:**  
Desarrollar en C++ los siguientes programas:  
(Se recomienda utilizar Funciones, Tipos definidos por el usuario y Enumeradores)

1. Resolver y Realizar en c++   
   1. Una aplicación para un restaurante que tiene: primer plato, plato principal y postre, toma pedidos de los mozos. Estos envían la comanda codificada por número para cada plato. El cocinero recibe el pedido en formato de texto con el nro de mesa, nombre mozo, fecha, hora, primer plato, plato principal y poste. Se pide desarrollar una APP que pida la comanda y muestre por pantalla la Info solicitada.
   2. Un empresa de viaje nos pide que realicemos un programa para tomar reservas en avión, las mismas pueden ser clase Primera, Bussiness y económica, además del dni apellido y nombre, aeropuerto origen y destino:  
      Código Aeropuerto  
      1. Bahía Blanca-BHI

2. Buenos Aires Aeroparque-AEP

3. Buenos Aires El Palomar-EPA

4. Buenos Aires Ezeiza-EZE

5. San Carlos de Bariloche-BRC

6. San Fernando del Valle de Catamarca-CTC

7. Comodoro Rivadavia-CRD  
El usuario introduce los datos personales, la clase y el origen – destino, mostrando el resultado por pantalla.

* 1. Una Verdulería nos pide que armemos el pedido para el mayorista para lo cual deberemos pedir al usuario el código proveedor, razón social, fecha de solicitud y fecha de entrega. Las frutas estarán enumeradas (tomar solo 6 a modo de ejemplo), mostrando los datos por pantalla.

Lic. Oemig José Luis.

1. Relacionar los conceptos de Tipo, Variable, Valor y Descripción, dar ejemplos.

**Tipo**: un tipo de dato es un atributo de los datos que indica al compilador la clase de datos que va a manejar, es decir, el conjunto de posibles valores que puede tomar el dato y las operaciones que tiene permitido realizarse con dicho dato.

**Variable**: es un determinado sector de memoria que tiene un nombre y un tipo.

Las ***variables*** se utilizan para poder manipular datos en C++. Para utilizar variables éstas se deben ***declarar***, esto es, asociar el un *nombre* con un *tipo*. Esto no es más que **describirle** al compilador qué cosas puede haber en esa variable (el tipo de datos que puede almacenar y qué operaciones se pueden hacer con ella). En la **definición** de una variable se asocia un identificador con un tipo y se asigna espacio físico en memoria (declaración + iniciación), es decir, se inicializa y se le asigna un **valor** a la variable.

**Ejemplos:**

int a; // Se está **declarando** la variable ‘a’, asignándole el **tipo** entero.

double b = 2.5; // Se está **definiendo** la var. ‘b’, dándole el **tipo** double

// y asingándole un **valor**

1. ¿Qué se entiende por Promoción y Conversión de tipos?

Cuando un operador binario relaciona dos operandos de distinto tipo, el valor del tipo *menor* se dice que ***se*** ***promociona*** al tipo *mayor* (se convierte al tipo mayor). El resultado tendrá el tipo *mayor*.

1. Explicar y ejemplificar el concepto de Conversión Segura y no segura

Se da un caso de ***conversión segura*** cuando se pasa de un tipo menor a un tipo mayor.

**Ejemplo**: short int 🡪 int 🡪 long int 🡪 …

En el caso de querer pasar de un tipo mayor a un tipo menor se está ante un caso de ***conversión insegura*** (con la consecuente posible pérdida de información)

**Ejemplo:**

int entero = 1234;

char caracter;

caracter = entero; // Conversión no segura

1. ¿Qué significa “Cast” o “Castear en castellanización”? dar ejemplos.

Existen dos tipos de “casting” (“castear” o moldear): El casteo implícito se produce en operaciones de asignación, paso de argumentos o cuando un operador vincula dos operandos de distinto tipo (como ya se estudió en la Unidad 2.2).

En el punto anterior se vio que, en ocasiones, esta conversión de tipo puede ser *no segura*. En lugar de depender de la coerción implícita de tipos, C++ permite hacer una conversión explícita de un tipo de dato a otro, a criterio del programador siempre y cuando estos tipos sean compatibles. De hecho se recomienda usar un modelo explícito de tipos para mostrar que la conversión de tipos es intencional.

En C++, la operación de moldeo viene en dos formas:

intVar = int(floatVar); // Notación funcional

intVar = (int) floatVar; // Notación de prefijo. Se requieren paréntesis

**Ejemplos**:

cout << double(a) / b; // Notación funcional

cout << (double) a / b; // Notación de prefijo

1. ¿Para qué casos usaría un tipo Definido por el Usuario? ¿En qué caso lo usaría?

Podría definir un nuevo tipo de dato para que el código gane en legibilidad y para poder expresar la solución (programa) en los mismos términos que el problema que se nos ha planteado. Por ejemplo, si necesito registrar las notas de mis alumnos, podría definir el tipo de dato tNotaAlumnno:

*typedef int tNotaAlumno;*

*notas tNotaAlumno;*

en lugar de:

*int notas;*

De este modo, no sólo es más legible; si en un futuro decido que es mejor que la nota sea un real en lugar de un entero sólo debería cambiar *una* línea de código: la primera. Así:

*typedef double tNotaAlumno;*

y el resto del código se mantendría intacto, mejorando así el mantenimiento del mismo.

1. ¿Qué es un tipo de Dato enumerado?, ¿en qué casos los utilizaría?

Un ***tipo de dato enumerado*** es un tipo de dato definido por el usuario cuyo dominio es un conjunto ordenado de valores literales expresados como identificadores.

Los utilizaría en caso de querer describir más fielmente la estructura subyacente de ciertos problemas. Por ejemplo al describir meses, días de la semana, estaciones del año y otros.

1. Explicar la sentencia que compone la Definición e Inicialización del Tipo enumerado con   
   un ejemplo, describiendo cada una de sus partes.

**La sintaxis para su uso es:**

*enum { símbolo1, símbolo2, ..., símboloN }*

*enum.* Nombre reservado por C++ para definir una enumeración.

*Nombre.* Cualquier nombre permitido que se le quiera dar a la enumeración.

*enumerador1, etc.* Lista de **enumeradores**.

**Ejemplo**:

enum dias { LUNES, MARTES, MIERCOLES, JUEVES, VIERNES, SABADO, DOMINGO };

dias dia;

dia = LUNES;

cout << dia << endl; // Salida: 0

Esta declaración crea un nuevo tipo de datos denominado ***dias***. Mientras que Typedef sólo crea un sinónimo para un tipo existente, un tipo de enumeración como ***dias*** es un nuevo tipo y es distinto de cualquier tipo existente

Los **enumeradores** son ordenados, en el sentido de que LUNES < MARTES < MIERCOLES... etc. Los valores en un tipo de enumeración se representan también internamente como enteros. Por omisión, el primer enumerador tiene el valor entero 0; el segundo tiene el valor 1, etcétera.

Se podría mejorar la *legibilidad* del código anterior mediante el uso de typedef del siguiente modo:

typedef enum { LUNES, MARTES, MIERCOLES, JUEVES, VIERNES, SABADO, DOMINGO } dias;

dias dia;

dia = LUNES;

cout << dia << endl; // Salida: 0

1. Los valores de tipo enumerado pueden castearse:

Ejemplo: Si al código anterior le agregamos las líneas:

int i = int(dia); // casteo explicito

cout << i << endl; // salida 0 (al igual que en el caso anterior)

1. Las constantes simbólicas se enumeran a partir de 0 (cero).
2. Explicar la siguiente “Casting Implícito” -> int num=tortuga dentro de un programa que use enumeradores.

Justamente, la variable *num* tomará como valor la ubicación de tortuga en la enumeración menos uno. De hecho, los valores en un tipo de enumeración se representan también internamente como enteros.

Podría incluso hacerse un casteo implícito como *double num = tortuga;*