Almacenamiento

Ya hemos visto que para poder operar con los datos, previamente hemos debido reservar unos espacios de memoria en los que almacenarlos. A esto es a lo que llamamos **estructuras de almacenamiento de datos.**

Para que los datos puedan ser correctamente tratados, estas estructuras deben ser definidas mediante:

- Identificador: nombre que le damos al dato durante el programa para referirnos a él.
- Tipo: naturaleza y rango de valores que puede almacenar.
- Valor: contenido concreto del dato en ese momento.

En este curso introductorio vamos a conocer dos estructuras únicamente, variables y vectores (también llamados arrays), si bien solo vamos a deternos en profundidad en la primera.

Variables y constantes

Son espacios que almacenan **un único DATO** de diferente naturaleza, y al que se le asocia un nombre que lo identifica. Los datos que pueden almacenar son:

- Numéricos: pueden ser enteros (int) o reales (float), según si aceptan o no decimales, generalmente con notación anglosajona donde el decimal se marca con un punto.
- Caracteres (char): letras o signos tipográficos. Se definen entrecomillando el carácter con una comilla simple.
- Cadenas de caracteres (string): palabras o frases. Se definen entrecomillando el texto con doble comilla.
- Lógicos (boolean): admite los valores de VERDADERO o FALSO.

Las variables, como su nombre indica, son estructuras de datos que van a tomar diferentes valores a lo largo de la ejecución del programa, mientras que las constantes adoptan el mismo valor durante todo el programa (Por ejemplo el número PI, o el porcentaje del IVA).

Es muy importante seleccionar adecuadamente el nombre de las variables y constantes para que ayuden a la comprensión de los datos almacenados. Generalmente las constantes se definen con nombres en mayúscula y las variables en minúscula.

Para conocer más de su funcionamiento, vamos a realizar un pequeño programa que solicite un precio, un porcentaje de oferta y a partir de un porcentaje de IVA (Constante) muestre un

mensaje con el precio total a pagar. Le llamaremos CALCULADORA DE REBAJAS.

Pasos 1 y 2: Análisis y diagrama de flujo del programa *Calculadora de Rebajas*

Los elementos implicados serán:

- **Inicio y fin** de programa.
- **Salidas**: Solicitar precio original y porcentaje de oferta, y mostrar el precio final.
- **Entradas**: Precio original, porcentaje de oferta.
- **Almacenamiento**: precio original (número real porque puede ser decimal), porcentaje de oferta (número entero porque usaremos el valor en tanto por ciento), precio final (número real), porcentaje de IVA (constante), y opcionalmente el mensaje a mostrar (constante)
- **Procesamiento**: sumas, multiplicaciones y divisiones.

El diagrama de flujo sería:

image-1657550591499.png

A las variables de igual forma también se les podrían asignar valores iniciales, aunque vayan a ser modificados a lo largo del programa. Si no se hace, asumen como valor inicial el 0 o vacío.

Pasos 3, 4 y 5: Codificación, compilación y verificación del programa *Calculadora de Rebajas* con PSeInt

Como hemos visto en el ejemplo de Entradas, PSeInt es un programa menos "riguroso" con el código, y nos ha permitido utilizar variables sin haberlas definido previamente (como hacíamos con num1 y num2, variables que almacenaban los valores de los números introducidos)

No obstante, el procedimiento ordinario en todo programa es definir previamente las variables (y constantes) que vamos a necesitar, mediante la asignación de un **identificador** y un **tipo de datos** a almacenar. En PSeInt esto se realiza mediante la instrucción **Definir.** Al escribir dicha instrucción, la ayuda nos solicita el nombre de las variables que queramos definir, separadas entre comas.

image-1657559249851.png

Podemos también especificar el tipo de datos que van a contener, añadiendo después del nombre la instrucción **como,** que nos abrirá las cuatro opciones disponibles en PSeInt: **entero, real, carácter o lógico**, también llamado booleano. En el caso de las cadenas de caracteres, no permite su definición como tal, si bien veremos que mediante asignación podemos almacenarlas en variables de igual forma.

image-1657559396197.png

Realizamos este proceso tantas veces como necesitemos, por el tipo de datos que hemos definido en el problema. En el caso de las constantes, les asignaremos su valor (21% en el caso del IVA y el texto del mensaje que queramos)

Para almacenar un valor en una variable o constante se utiliza el comando **Asignar**, de la ventana de la derecha.

image-1657630542224.png

También se pueden escribir directamente el signo menor y el guión mediante el teclado.

image-1657629506076.png

En el momento en que definimos variables y constantes en el programa, si hacemos clic en la pestaña situada a la izquierda llamada **Lista de Variables** se nos abre una ventana con las variables definidas.

image-1657629840670.gif

Una vez definidas las variables que van a contener la información necesaria, solicitaremos y almacenaremos los valores del precio inicial y el porcentaje de rebaja.

image-1657630216896.png

Por último, operaremos con ellos mediante operaciones aritméticas (ver siguiente punto) y asignaremos el valor obtenido a la variable precio_final que será la que mostraremos por consola.

image-1657630645655.png

Como se puede apreciar en la última fila del programa, la instrucción **Escribir** permite la concatenación de diferentes tipos de datos sin más que separarlos entre comas.

Ya solo nos quedará comprobar el correcto funcionamiento del programa dándole a **Ejecutar**. Debido a la mayor longitud de este programa puede ser un buen momento para comprobar como funciona la orden **Ejecutar paso a paso.**

image-1657630951667.gif

En el paso de depuración de errores y verificación del programa puedes probar a introducir errores posibles, para ver cómo reaccionaría el programa: por ejemplo meter los decimales con una coma,

o un porcentaje de descuento decimal. En programación es fundamental adelantarse a equívocos o posibles confusiones de la persona usuaria, para tener prevista una respuesta que no bloquee el programa o prevenirlas mediante mensajes de aviso.

Pasos 3, 4 y 5: Codificación, compilación y verificación del programa *Calculadora de Rebajas* con Scratch

En Scratch, las estructuras de almacenamiento de datos se encuentran en el bloque **Variables**. Desde allí podremos crear tantas variables como necesitemos, y también asignarles el valor deseado mediante el bloque correspondiente.

image-1657631516502.png

Al crear una variable lo primero que nos solicita aparte de su identificador es saber si se define como **local** (solo de este objeto) o como **global** (común para todos los objetos) En este curso solo desarrollamos programas de un mismo objeto por lo que nos daría igual.

image-1663515981704.png

Una vez definidas las variables necesarias pasaremos a crear el código en el objeto utilizando **sensores**, **apariencia** y los **operadores** aritméticos y de concatenación que necesitemos, y que veremos con más detalle en el siguiente apartado. El código en bloques quedaría de la siguiente forma:

image-1657635736663.png

Pruébalo aquí:

https://scratch.mit.edu/projects/750474716/embed

Estructuras complejas de almacenamiento de datos

Aunque no es objeto de este curso, en programación se pueden almacenar **conjuntos de datos** siempre que sean **del mismo tipo** en otro tipo de estructuras complejas, los llamados **arrays** que en español se traducen como vectores o matrices, y en PSeInt se denominan **dimensiones**.

Para más información sobre cómo utilizar dimensiones en PSeInt se puede consultar el siguiente video.

https://www.youtube.com/embed/889nGzmU2yA

En Scratch solo se contempla la utilización de **listas** que serían series ordenadas de datos del mismo tipo, y que se encuentran disponibles dentro del mismo bloque **Variables**.

image-1657636178529.png

Financiado por el Ministerio de Educación y Formación Profesional y por la Unión Europea - NextGenerationEU

logo.png

Revision #22
Created 2022-06-05 14:46:41 CEST by Ana López Floría
Updated 2023-01-17 15:48:23 CET by Equipo CATEDU