



ENES
JURIQUILLA

Ingeniería Aeroespacial

204 - Fundamentos de Programación

Ing. Andrés David Flores Ferriz

Práctica 2

Solución de práctica
algoritmos

Actividad en clase 23 de 25 pts

Protocolo de práctica	57 de 75 pts
Portada	08 de 10 pts
Introducción	07 de 15 pts
Desarrollo	19 de 25 pts
Conclusiones	15 de 15 pts
Referencias	08 de 10 pts

CALIFICACIÓN TOTAL Entrega 80 pts

Equipo 1

Grupo: 2002

Semestre 2

Paul Sebastian Gutierrez de Loza

Emilio Laver Aguilar

Marianangel Ramos Garzúlez

Nro. Cuenta

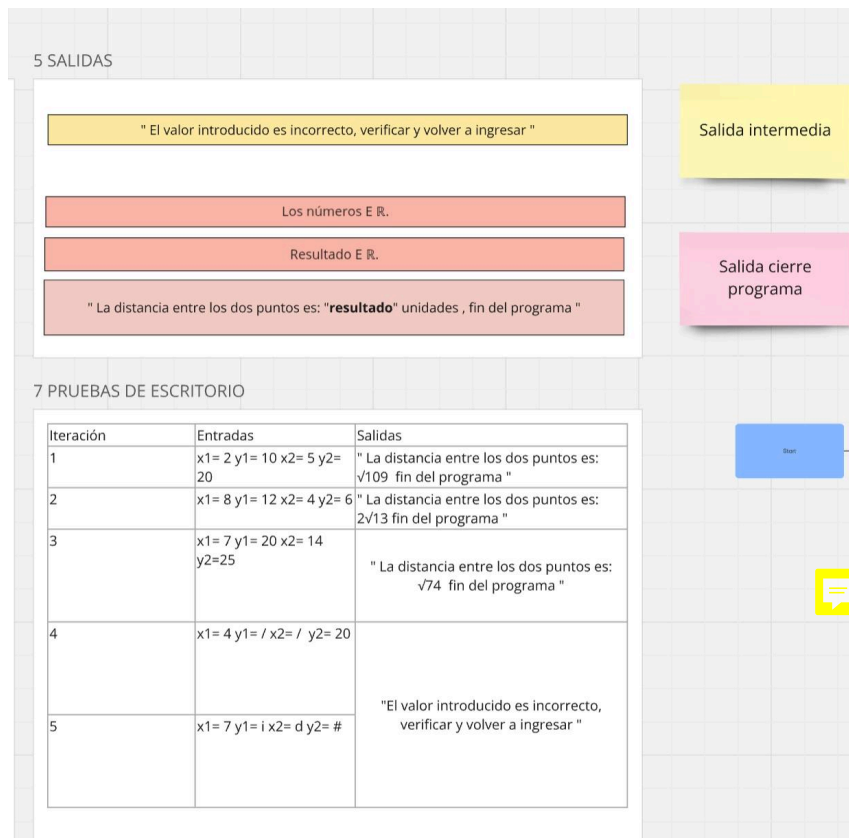
425037061

425117677

425062412

Fecha de entrega: 14 - feb - 2025

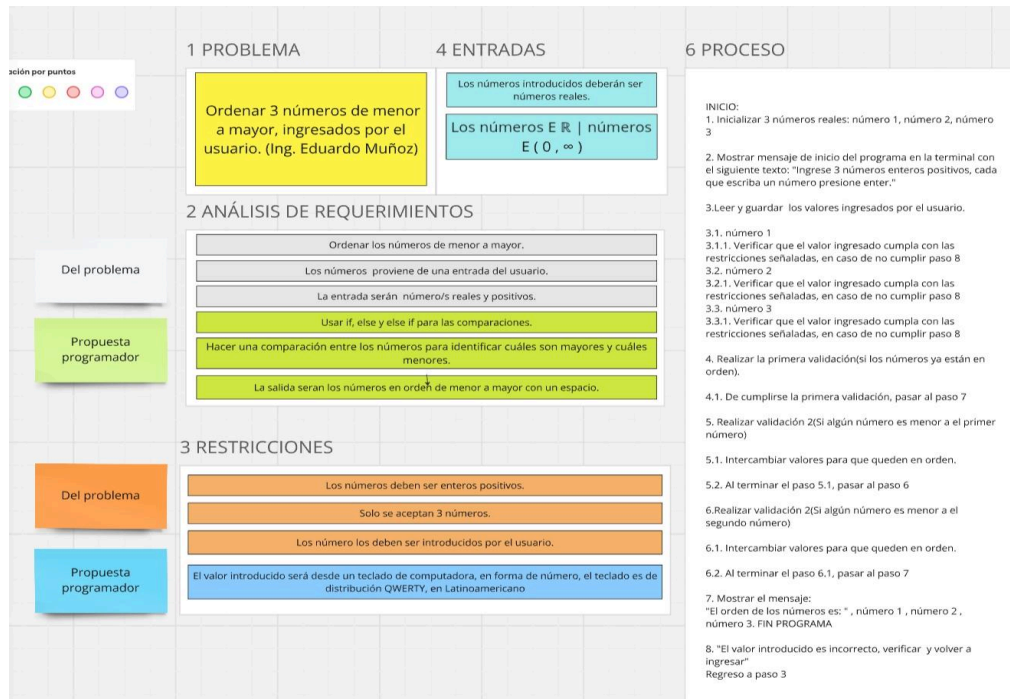
En el 4 se describen cuáles son las entradas del sistema en las que se especifica que deben de ser números enteros pertenecientes a los números reales desde el 0 hasta el infinito. El 6 es el paso a paso de lo que el programa va a hacer explicado con palabras, explicando cuál será el inicio, en caso de cumplir o no cumplir con lo que se necesita que pasará y finalmente cuál será el resultado de ese problema.



El 5 habla sobre las salidas, esto es lo que va a salir una vez que se tenga el resultado del programa diciendo "La distancia entre los dos puntos es: "# unidades, fin del programa ". El resultado deberá ser un número entero perteneciente a los reales

En el 7 son pruebas o ejemplos de cómo debería de suceder el proceso de entradas y salidas, en este se toman varios ejemplos de lo que podría pasar en caso que entren de manera correcta los valores o que debe de hacer si los valores ingresados no son compatibles con el programa.

Reto 2



En el 1 se describe el problema que escogimos que serían ingresar 3 números y que estos sean ordenados de mayor a menor.

En el 2 de análisis de requerimientos en el apartado de lo que necesita el problema son los 3 números dados por el usuario, el programa deberá ordenar los números de mayor a menor y con la condición que los números ingresados podrán ser números enteros pertenecientes a la familia de los reales y deberán ser positivos. Por parte del programador deberá usar if, else, y else if para poder comparar los números y lograr acomodarlos de menor a mayor y de salida deberán salir ordenados con un espacio.

En el 3 las restricciones por parte del problema pide que sean números enteros positivos, solo pueden ser 3 números y los números deben de ser introducidos por un usuario. Por parte del programador deben de ser números arábigos y en un teclado tipo QWERTY.

En el 4 son las entradas que deben ser que los números introducidos deberán ser números reales y estos deben ser pertenecientes a los reales.

El 6 explica todo el proceso que deberá estar en el programa para poder resolver el problema de manera correcta

5 SALIDAS		
" El valor introducido es incorrecto, verificar y volver a ingresar "		
numero1<numero2<numero3		
"El orden de los números es: " , numero 1 , numero 2 , numero 3.		
7 PRUEBAS DE ESCRITORIO		
Iteración	Entradas	Salidas
1	5 7 6	"El orden de los números es: "5, 6, 7
2	98 51 102	"El orden de los números es: "51, 98, 102
3	8 15 4	"El orden de los números es: "4, 8, 15
4	5, -4, 23.5	"El valor introducido es incorrecto, verificar y volver a ingresar "
5	8.2, 5.9, 7.6	

En el 5 son las salidas y en caso de que haya un error en los números introducidos deberá de salir una leyenda diciendo “El valor introducido es incorrecto, verificar y volver a ingresar”, en caso de que todo esté perfecto va a salir el resultado ya ordenado de menor a mayor con la leyenda: “El orden de los números es:”#<#<#”

En el 7 se ponen ejemplos de cómo debe de funcionar el programa en caso de que todos los números se hayan ingresado de manera correcta y en caso de que no como corregirlo

Conclusiones

Hacer el algoritmo para ordenar los tres números fue más difícil que obtener la distancia entre dos puntos. Para la distancia simplemente es obtener los cuatro valores, checar una por una que las coordenadas sean válidas y usar la fórmula para la distancia entre dos puntos.

Para ordenar los tres números fue algo más complicado, lo que propusimos es ir comparando los números, en caso de que ya estuvieran en orden darlos de regreso al usuario, si no ir intercambiando hasta que lo estén. La práctica fue de mucha utilidad, hacer esto antes de hacer el código permite ver todo lo que debe y no debe de hacer el código y poder corregir errores antes de cometerlos dentro del código.

Comentarios

Paul: Me gusto mucho el tema, el saber como organizar de manera óptima la información y el crear diagramas y algoritmos para posteriormente programarlos es muy útil.

Lara: Creo que poder organizar la información de esta manera hará más fácil hacer códigos complejos, para saber discernir entre información o acciones redundantes y que el código sea más corto y eficiente.

Gel: Me entretuvo bastante al buscar la forma en cómo se debía de acomodar para que fuese claro y preciso. Si no se hacía de esa manera caía en los errores y quedaba de manera muy general, la verdad si se veía feo cuando se hace general, por lo tanto concluyo que aprendimos a ser más específicos y no dejar las cosas sin aclarar.

Referencias

Runestone Academy. (s.f.). El ordenamiento burbuja. En *Python para la educación computacional*. Runestone Interactive.

<https://runestone.academy/ns/books/published/pythoned/SortSearch/EIOrdenamientoBurbuja.html>

