**Do-Loops y escribiendo en archivos Fortran**

**Introducción**

**Do-Loops**

En general, un programa FORTRAN ejecuta los comandos en el orden en el que se escribieron, uno a la vez. Sin embargo, frecuentemente hay situaciones en las que esto es demasiado simple para lo que uno quiere hacer. A veces es necesario repetir una misma operación muchas veces con ligeras variaciones, y otras veces hay operaciones que solo 16

deben realizarse si se cumple alguna condición. Para este tipo de situaciones existen los llamados “comandos de control del programa” y que caen en dos tipos básicos: “loops” y condicionales.

Un “loop” es un conjunto de instrucciones que deben realizarse muchas veces y tiene la forma estándar:

1. do i = start,end,increment
   1. comando 1
   2. comando 2
   3. comando 3
   4. ...
2. end do

Todos los comandos que están contenidos entre la línea do i=... y la línea end do se repiten varias veces, dependiendo de los valores de start, end, increment que deben ser enteros. la primera vez, la variable i toma el valor start. Al terminar, el valor de i se incrementa en increment, si el valor final es mayor que end el loop ya no se ejecuta de nuevo y el programa continúa en el comando que siga después de end do, de lo contrario se vuelven a ejecutar los comandos dentro del loop hasta que se obtenga i>end. Nota: El indentar los comandos dentro del loop no es necesario (FORTRAN ignora los espacios extra), pero es una buena idea pues hace más fácil identificar visualmente donde comienza y termina el loop.

**Procesamiento de archivos**

Las aplicaciones que manejan conjuntos de datos muy grandes, es conveniente que almacenen los datos en algún archivo en disco o algún otro dispositivo de memoria auxiliar.

Antes de que Fortran pueda usar un archivo, debe abrirlo, asignándole una unidad. La sintaxis general para abrir un archivo es:

OPEN ("lista\_open")

donde lista\_open puede incluir varias cláusulas separadas por comas, colocadas en cualquier orden.

En esta actividad aplicaremos el concepto de [do-loops](https://www.tutorialspoint.com/fortran/fortran_do_loop.htm), estructurar procesos repetitivos, así como también aprenderemos a leer entradas*y*escribir la salidade los datos, de un archivo de texto a otro, utilizando la instrucción de [open de Fortran](https://www.tutorialspoint.com/fortran/fortran_file_input_output.htm).

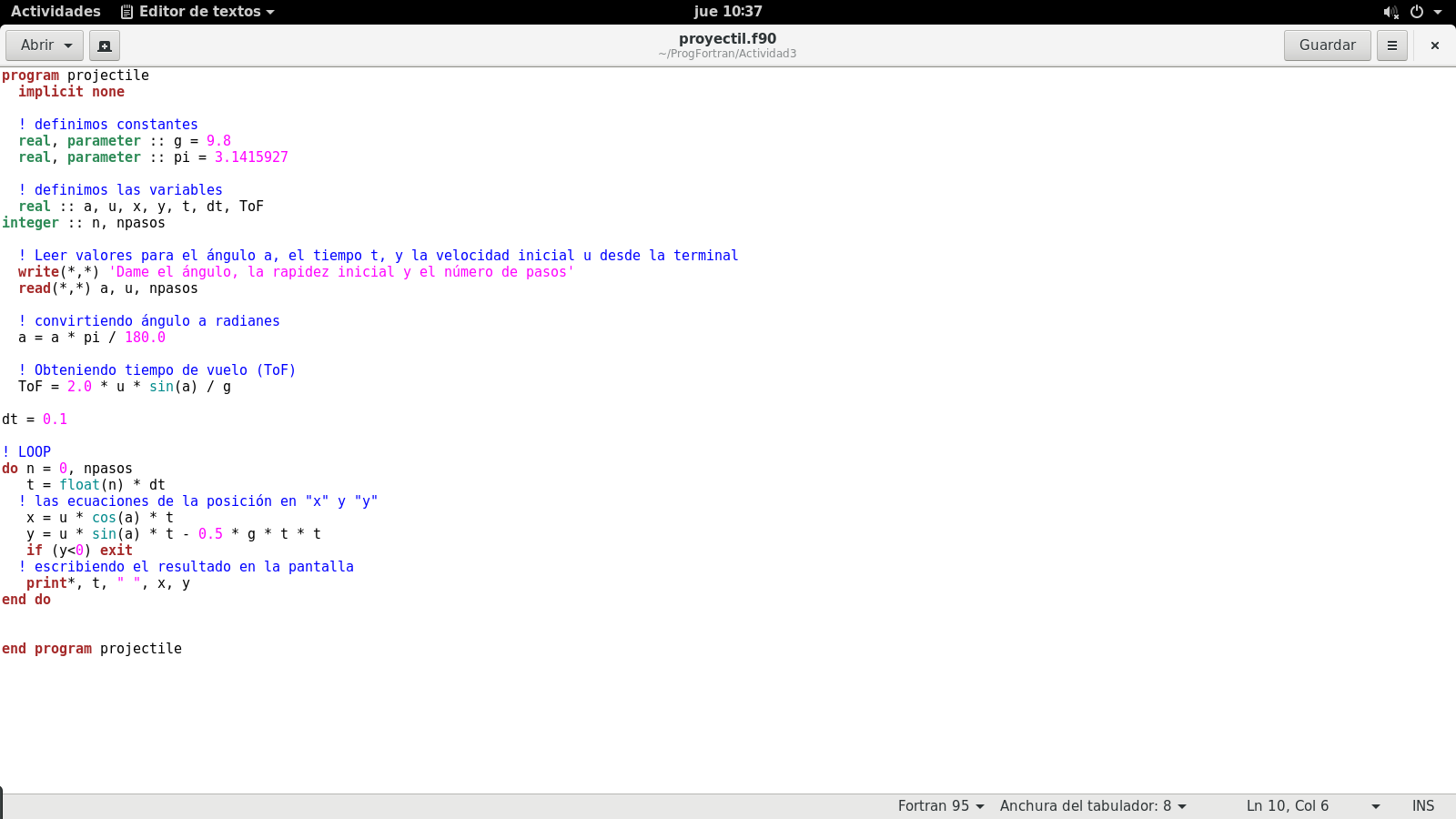
En forma paralela, también aprenderemos a utilizar el gratificador [Gnuplot](http://www.gnuplot.info/), para intentar reproducir la gráfica que aparece al inicio de la actividad y tomada del artículo de la Wikipedia sobre el [movimiento de proyectiles](https://en.wikipedia.org/wiki/Projectile_motion).

**Descripción de la actividad**

1. Elabora un programa que lea la velocidad inicial u y el tiempo total de vuelo t.
2. Estructura tu primer[do-loop,](https://www.tutorialspoint.com/fortran/fortran_do_loop.htm) para que el programa escriba en la pantalla la posición (x,y) del proyectil cada 0.1 segundos para un ángulo especifico.
3. Define un archivo "salida.dat" con la instrucción [open](https://www.tutorialspoint.com/fortran/fortran_file_input_output.htm), donde se escribirán las posiciones (x,y), para cada instante de tiempo.  Verifica posteriormente que el archivo contiene resultados coherentes.
4. Utiliza [Gnuplot](http://gnuplot.sourceforge.net/demo/), para grafícar el archivo "salida.dat". Desde una terminal ejecuta el programa gnuplot.  Dentro de la interfaz de Gnuplot, escribe: plot "salida.dat" with lines
5. Logrado lo anterior,  ahora incluye otra estructura loop que repita los cálculos anteriores, pero para 6 ángulos distintos: 15, 30, 45, 60, 75 y 90 grados. Los datos de las posiciones (x,y), se escribirán también al archivo "salida.dat"
6. Utiliza [Gnuplot](http://gnuplot.sourceforge.net/demo/) para traficar de nuevo el archivo "salida.dat". Busca en Internet cómo graficar varias curvas con Gnuplot y agregar etiquetas, para tratar de reproducir la figura de la Wikipedia.

**Resultados**

* Pasos 1 y 2



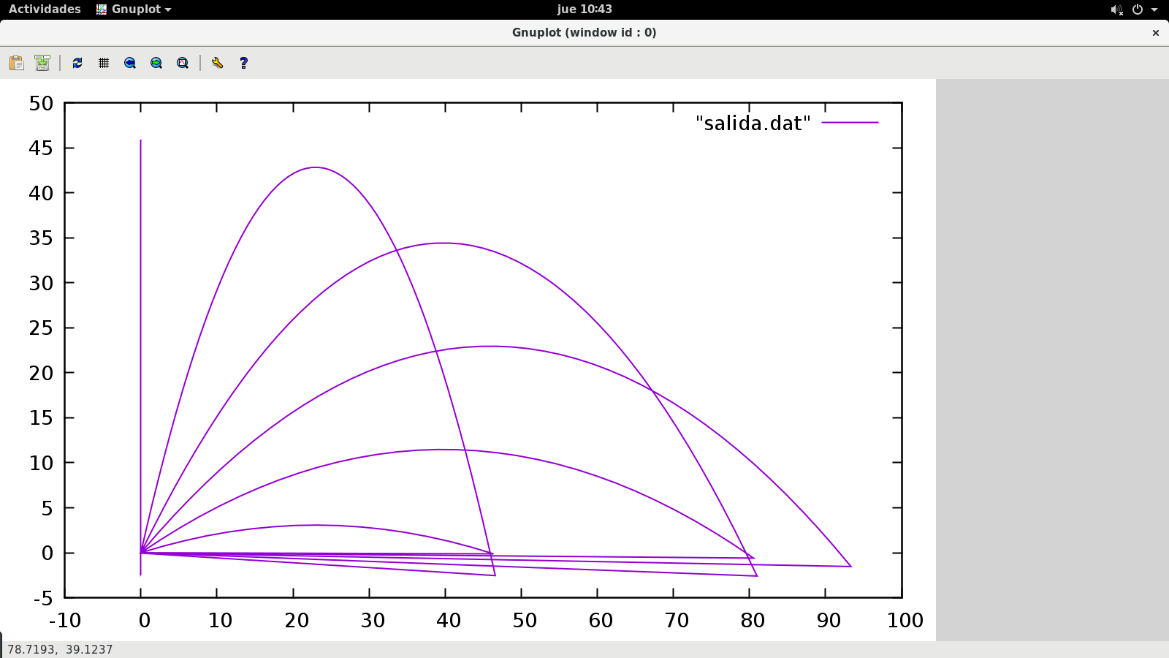
A partir del programa de la actividad anterior, se escribe un programa en Fortran con un primer do-loop al final del programa. Este loop se establece para obtener las coordenadas “x” y “y” en diferentes tiempos con intervalos de 0.1 segundos, hasta que el proyectil caiga al suelo.

* Pasos 3, 4 y 5

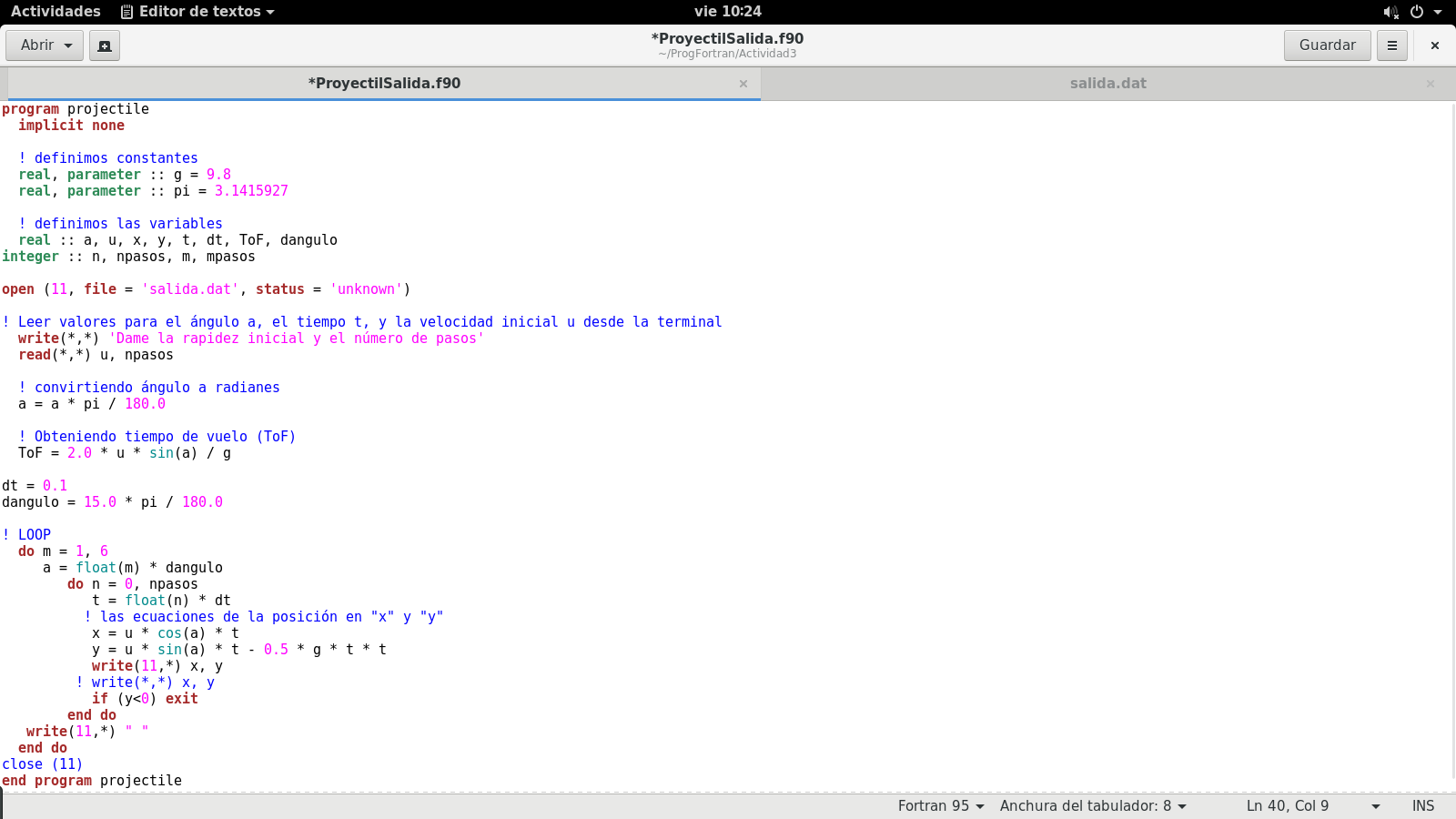


Se observa la misma sintaxis que en el paso anterior, con el añadido de que se abre un archivo y se escriben los datos de salida en él. Nótese que también se añadió otro loop para variar los ángulos iniciales del movimiento parabólico.

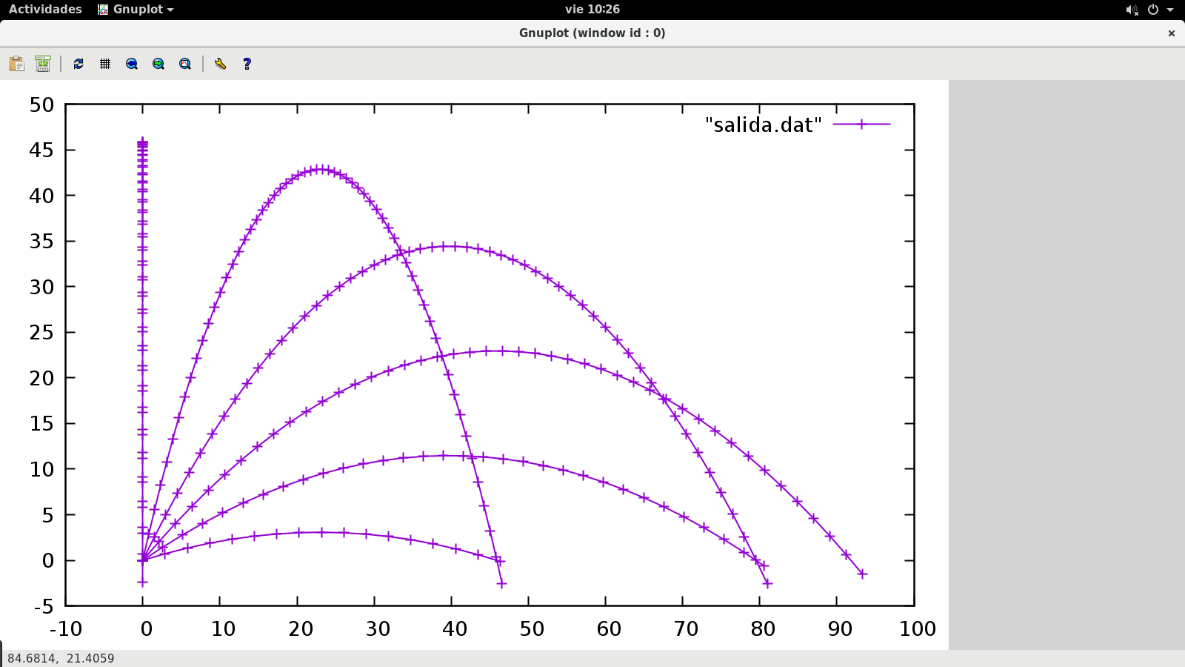
Al final del programa se pueden observar dos líneas que erróneamente fueron escritas en el programa, ya que esto causa que el último dato de posición en “x” y “y” se reescriba tanto en la pantalla de salida, como en el archivo “salida.dat”, lo cual nos arroja una gráfica como la siguiente:



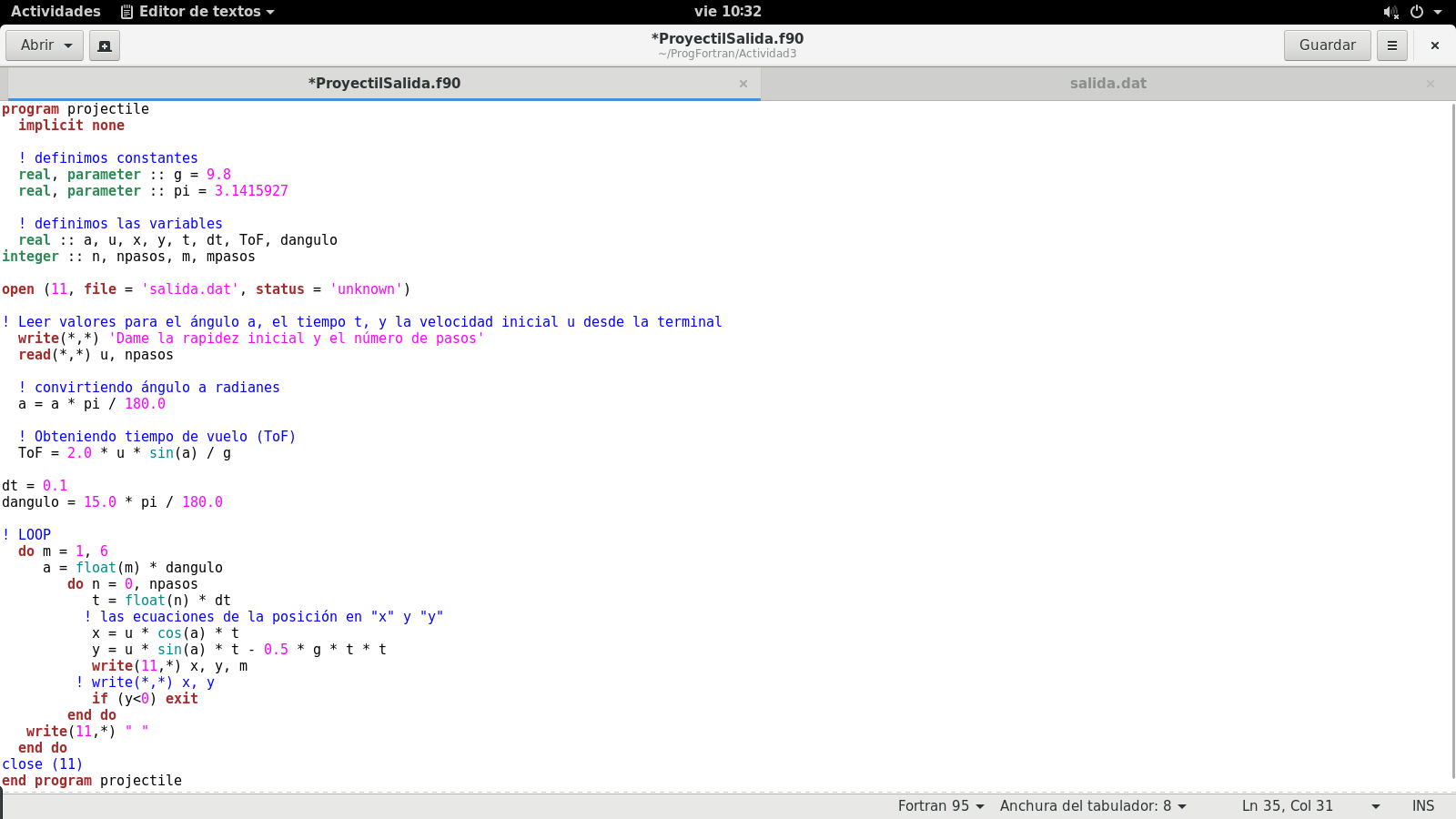
Ya que en el programa no se especifica ninguna separación entre los datos, GNUPLOT toma todos los datos juntos como en una sola gráfica, es por esto que la gráfica se ve continua para todos los tiros diferentes.



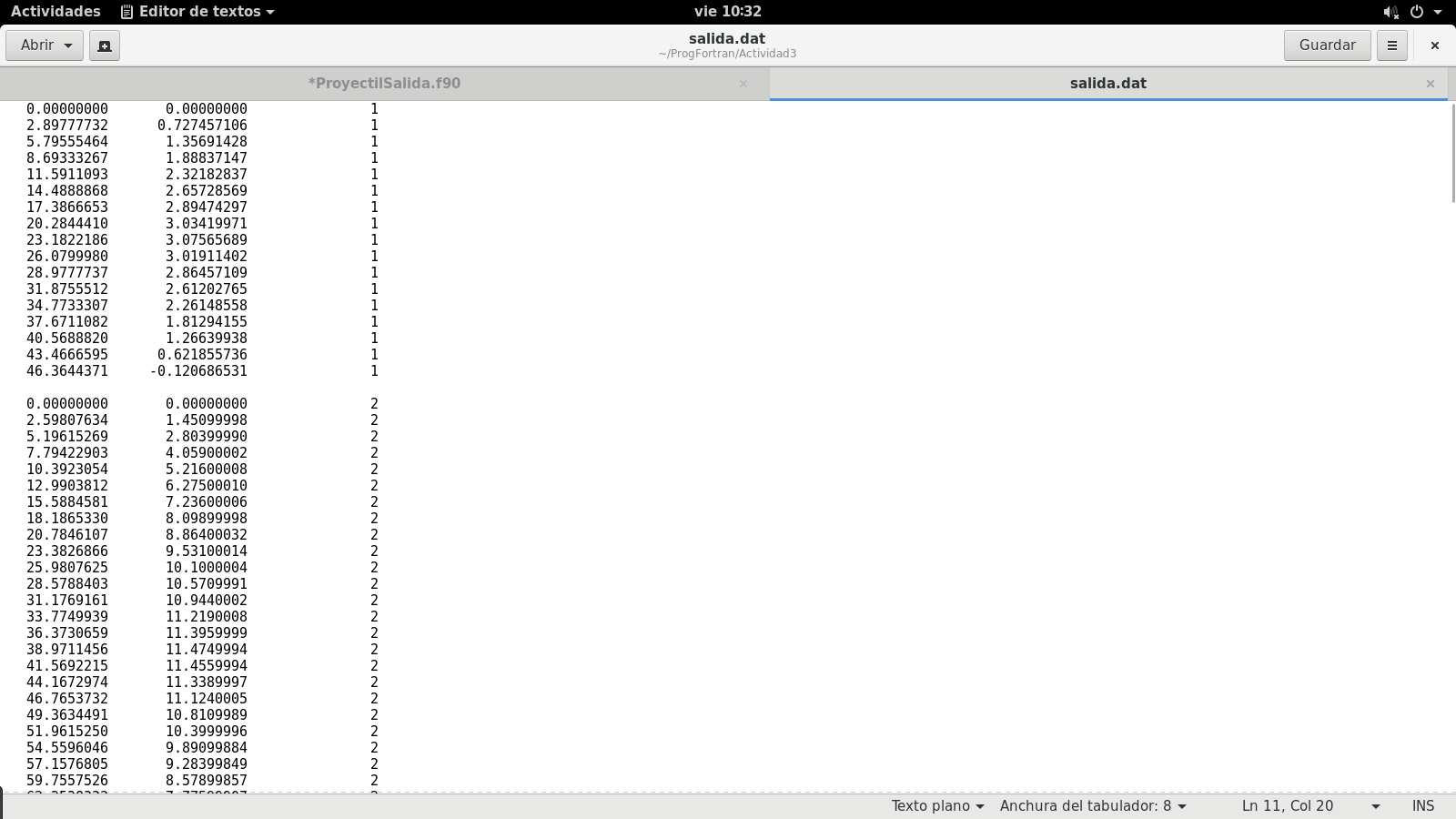
Para corregir los errores anteriores, se eliminó el comando de escribir fuera del loop, además se añadió un espacio cada vez que se cambie el ángulo inicial. Lo cual nos muestra un archivo de salida con los datos separados por cada ángulo, arrojando una gráfica más certera y separada por cada tiro parabólico simulado.



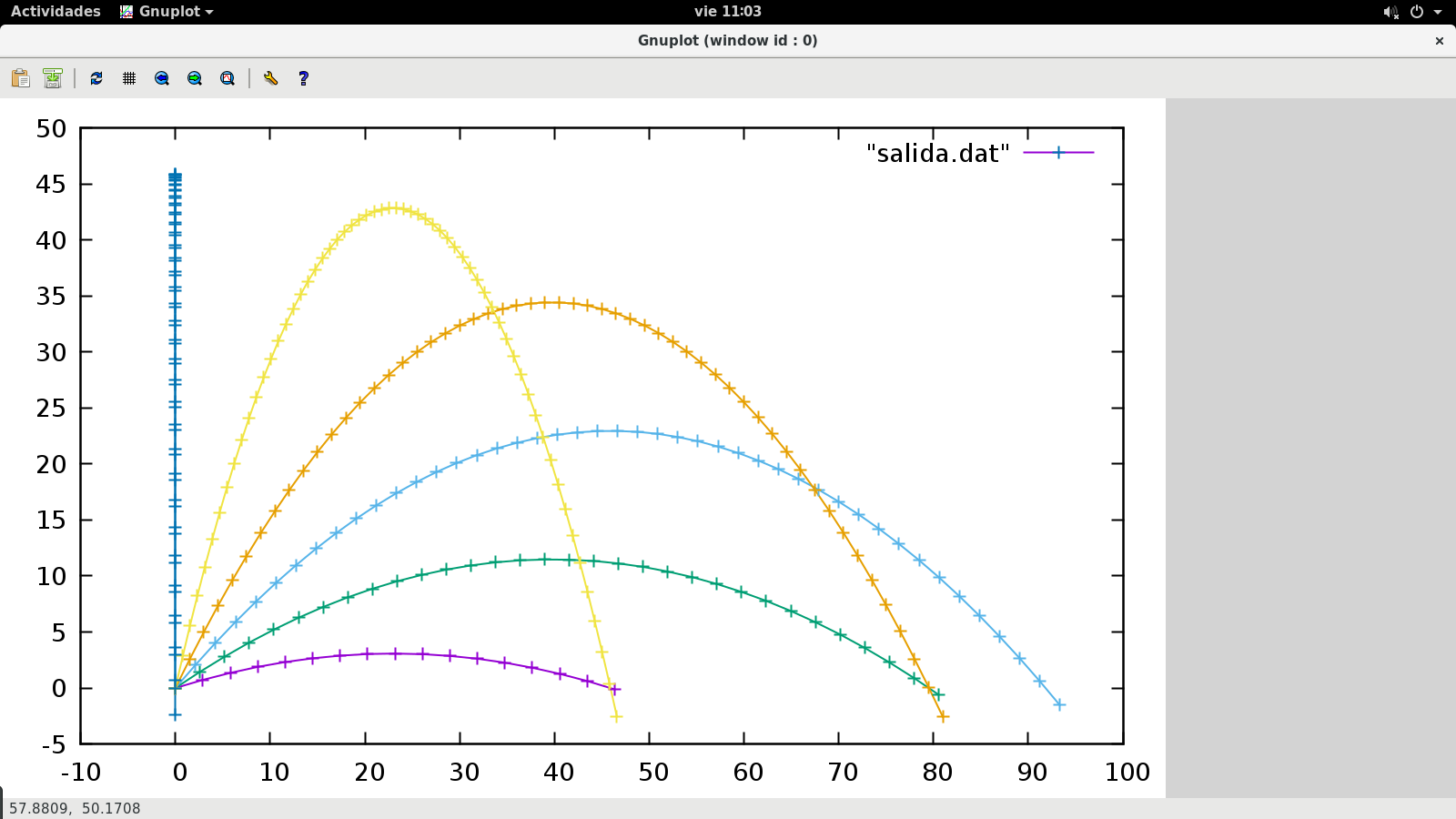
* Paso 6



Se añade ahora el término “m” al momento de escribir en el archivo “salida.dat”, esto con la finalidad de marcar cada uno de los tiros parabólicos y lograr distinguir sus gráficas de movimiento con diferentes colores.

Con la sintaxis dada a GNUPLOT, se le pide que grafique con puntos y líneas las diferentes gráficas y, además, que use la tercera columna de datos para asignar un color a las líneas, variando el color entre los diferentes números. Esta orden arrojará la siguiente gráfica:



**Conclusión**

Es importante entender la manera en que la sintaxis para Fortran trabaja, siguiendo un orden lógico de órdenes que se le dan al ordenador para lograr obtener el resultado deseado, los problemas de sintaxis pueden resolverse fácilmente al momento de compilar, ya que la misma terminal nos avisará si hay un error y dónde está el error a corregir, por lo que es más importante entender la lógica a seguir para que el programa funcione correctamente.

Al final de esta práctica y ajustando los errores que se tenían anteriormente, se obtiene una gráfica con un color asignado para cada uno de los tiros parabólicos simulados en el programa, concluyendo de esta manera con éxito la actividad 3.

**Bibliografía**

* <http://progfortran.pbworks.com/w/page/135226839/Actividad%203%20(2019-2)>
* Tutorialspoint. <https://www.tutorialspoint.com/fortran/fortran_do_loop.htm>
* Tutorialspoint. <https://www.tutorialspoint.com/fortran/fortran_file_input_output.htm>
* Introducción a Fortran. Miguel Alcubierre. Instituto de Ciencias Nucleares, UNAM. Abril 2005. <http://pelusa.fis.cinvestav.mx/tmatos/LaSumA/LaSumA2_archivos/Supercomputo/Fortran.pdf>
* LibroFortran. Formatos y archivos.

<https://ocw.unican.es/pluginfile.php/1904/course/section/2188/Curso-Fortran-7.pdf>