ddd Tips

This provides some helpful information about the Unix **ddd** debugger. There are other resources available with more detailed information:

gnu ddd manual

matloff ddd information

Contents

Prerequisites

- Your code must have been compiled with the **-g** compiler option. Example:

  `gcc -g -o` *`executableName programName`*`.c`

- ddd uses an x-window display. If you are using it remotely, you must have ubuntu (or a unix-emulation product such as cygwin) and started an x-window terminal window.

Launching ddd

ddd *executableName*

Overview of ddd

When ddd starts (it takes a while, if it takes longer than 30 seconds, see How can I fix the "GDB is Busy" error? .**Error! Bookmark not defined.**), it shows a Tip of the Day. These will be helpful for most students. It shows the initial DDD window (see Figure 1: Initial Display) which consists of the following areas:

Menu Bar	This is at the menu at the top of the window and includes File, Edit, View, etc.
Tool Bar	This is immediately below the menu bar and contains some important tools like Find and Break.
Data Area	At the beginning, this shows an empty grid area. It is used to show values of variables which you wish to closely examine.
Source File Area	This displays your source code.
Command Tool Area	This appears on the right side of the Source File Area and includes buttons:

Run	run the program.
Step	execute the current step. It will step into a called function.
Next	execute the current step, but do not step into a called function.
Cont	Continue execution until the next break or stdin.
Finish	continue execution until the finish of the current function.
Kill	Kill the execution of your program.

Console Area — This is at the bottom of the screen:

- debugger commands are shown/entered
- standard input is entered (depending on preferences)
- standard output is shown (depending on preferences)

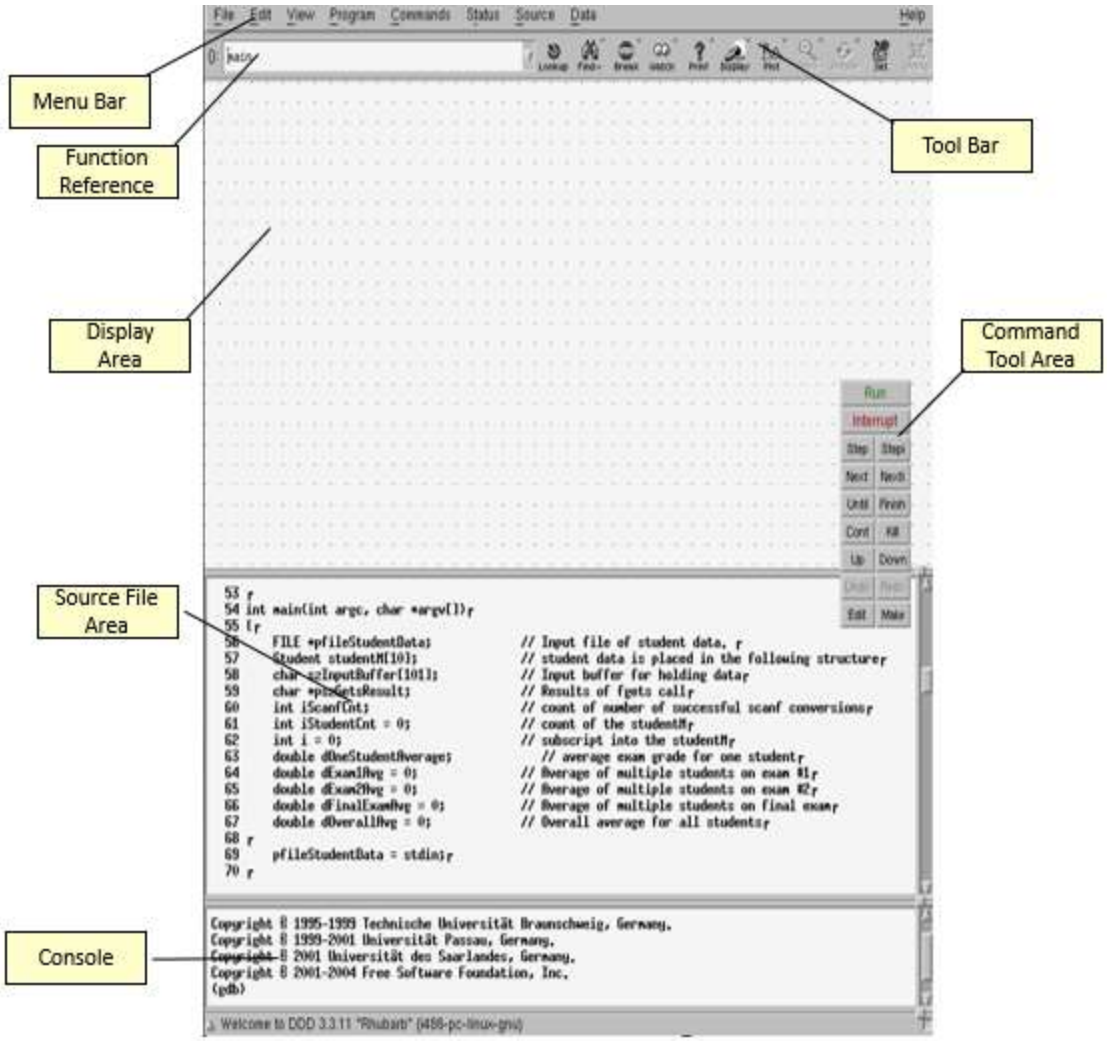


Figure 1: Initial Display

How can I tell ddd to show line numbers?

This is done via the **Source** menu item.

Source > Display Line Numbers

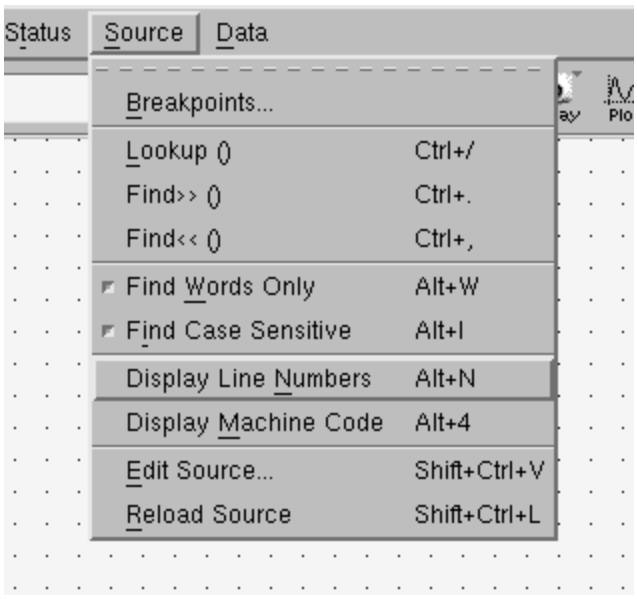


Figure 2: Showing Line Numbers

Running Your Program

Although there are multiple mechanisms provided to run your code within ddd, it is easiest to specify **program arguments** and how to handle **stdin** by using the **Program** menu item.

- If you are using stdin, it is easier to first tell ddd to use an Execution Window:

  Program > Run in Execution Window (you only have to do this one time)

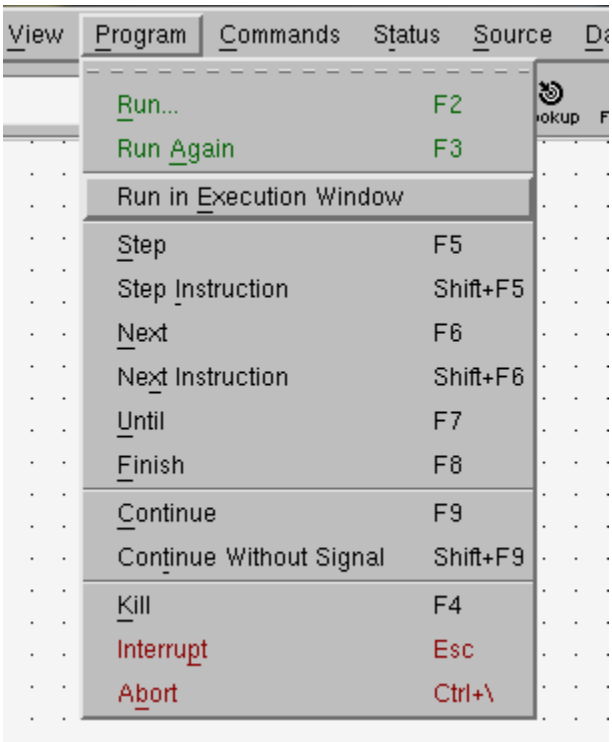


Figure 3: Running in an Execution Window

And then to tell ddd to run it:

Program > Run…

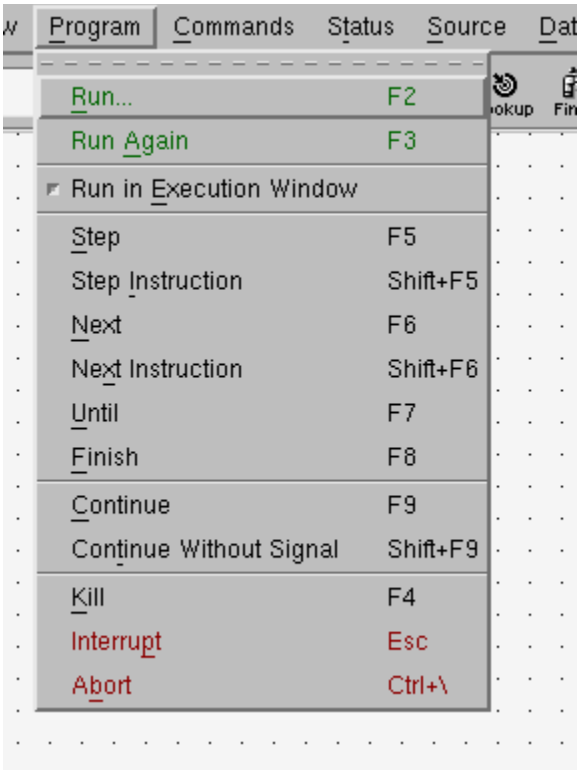


Figure 4: Run

- If you need to enter command line arguments for your program, use

  Program > Run…

  ddd will show a window which allows you to either select previously entered arguments or enter new ones

- If you want to redirect stdin to come from a file, use

  Program > Run…

  ddd will show a window which allows you to either select previously entered arguments or enter new ones. You can specify command line arguments, stdin redirection or both:

  args

  <inputFileName

  args <inputFileName

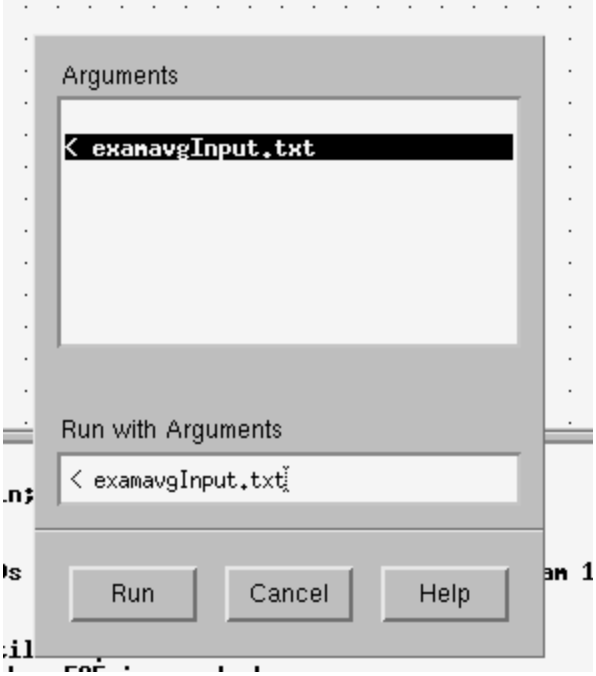


Figure 5: Redirecting stdin

What happens after run begins?

After **run**, your code will execute until whichever it first encounters

- program exit or program return
- a request for stdin from the keyboard
- a Breakpoint (which you must manually set)
- a Watch Point variable's value changes

How can I set breakpoints?

Click on a line number. Your right mouse button will give you a popup menu where you can set a breakpoint. When you set it, a stop sign will appear.

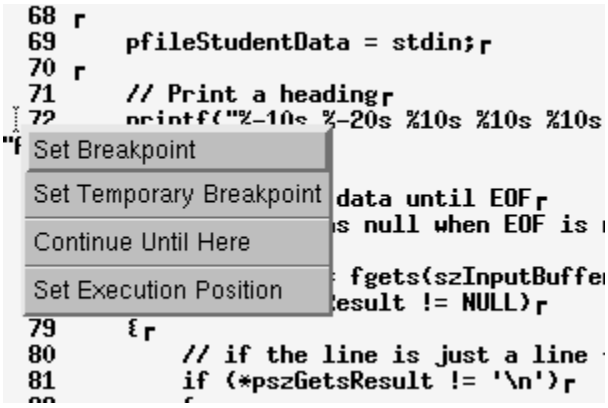


Figure 6: Setting a Break Point

When you run your code, if the breakpoint is encountered, execution will stop at it. The current point of execution is shown with a large arrow.

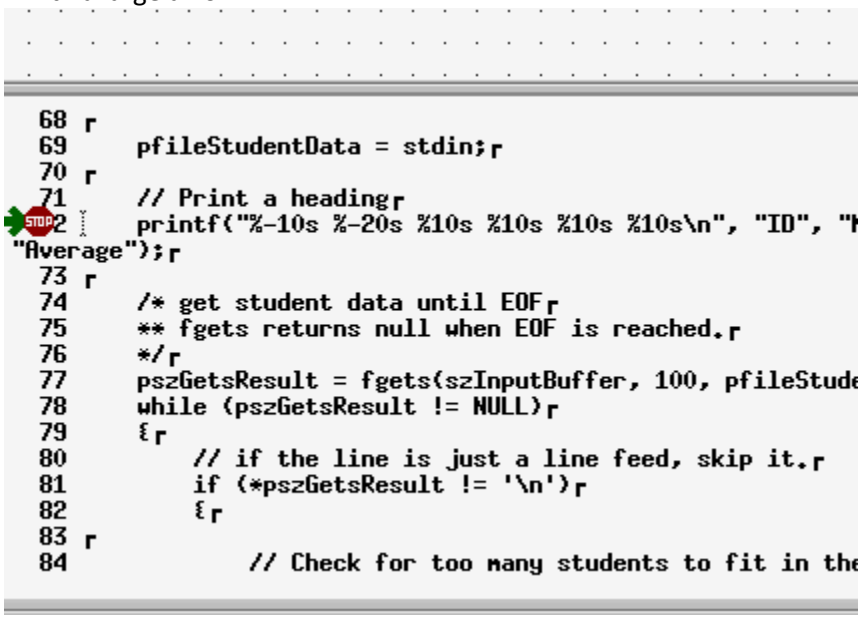


Figure 7: Program Breaking at a Breakpoint

How can I set a breakpoint in another source file?

Sometimes programs are separated into multiple source files. ddd starts by showing the **main** function. If you would like to set a breakpoint in a function named "insert" which is in another source file, simply click in the source function reference window on the tool bar. Replace "main" with your function name (e.g., insert). See figure 1 for location of the function reference window.

How can I remove a breakpoint?

Click on the line number containing the breakpoint. Your right mouse button will give you a popup menu allowing you to delete the breakpoint.

How can I look at the value of a variable?

For a simple variable, you can hover the mouse over the variable and its value will show during execution. For an array reference (e.g., `studentM[i].szStudentFullNm`), highlight the entire reference and then hover over that selected text.

You can also display a variable in the Display Area. Right click on a variable and it will be shown in the Display Area. This is useful to see how the variable changes as the code executes.

How can I look at an array or complex structure?

Arrays and complex structures are usually shown in the Display Area. Right click on the array and select **Display.** If your variable is a pointer, you might prefer to display what it references by selecting **Display*.**

The example in Figure 8 shows what is displayed for the following array:

```
typedef struct
    {
        double dExam1;                  // exam #1 score
        double dExam2;                  // exam #2 score
        double dFinalExam;              // final exam score
        char szStudentIdNr[7];          // Student Identification Nr
        char szStudentFullNm[21];       // Student full Nm
    } Student;
Student studentM[10];
```

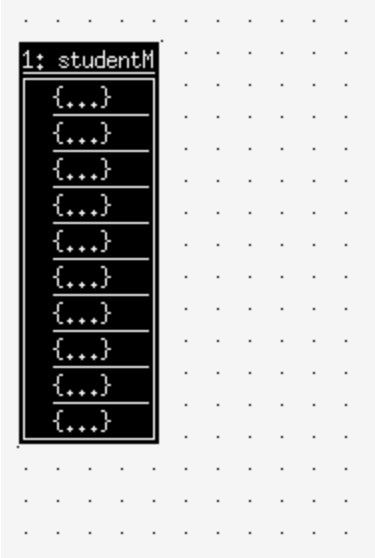


Figure 8: Display of studentM

To examine the contents of an element of that array, click on the element and then you can do any of the following on that element:

- right click menu and select **show all**. Figure 9 shows what would then be displayed for that element.
- right click menu, select **New Display**, select **Other…**, and a New Dependent Display window appears. It should show you which element you selected. Press the **Display** button in that dialogue window. Figure 10 shows the display.

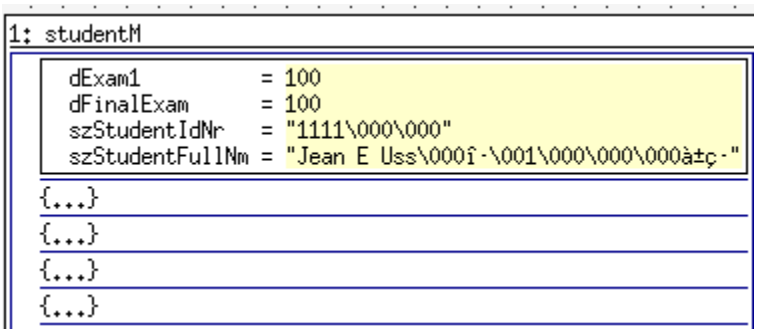


Figure 9: Showing the contents of studentM[0] using show all

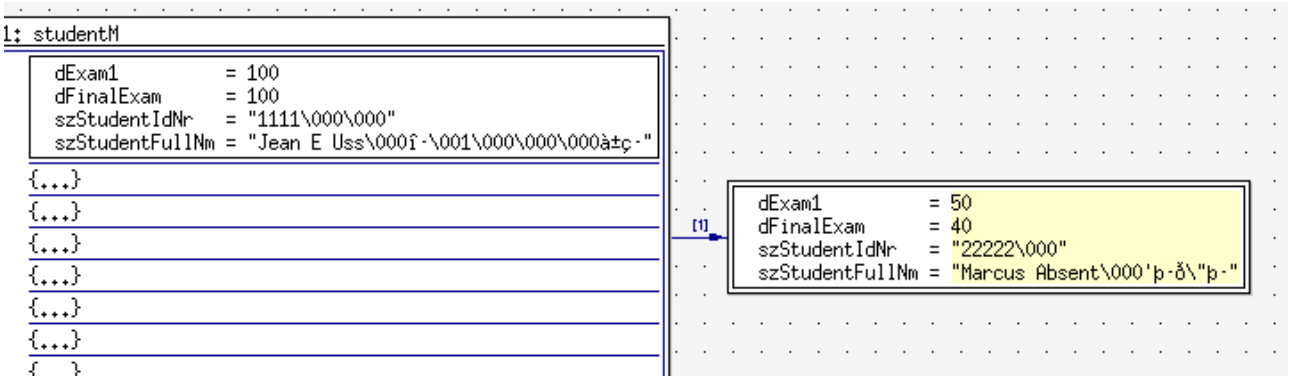
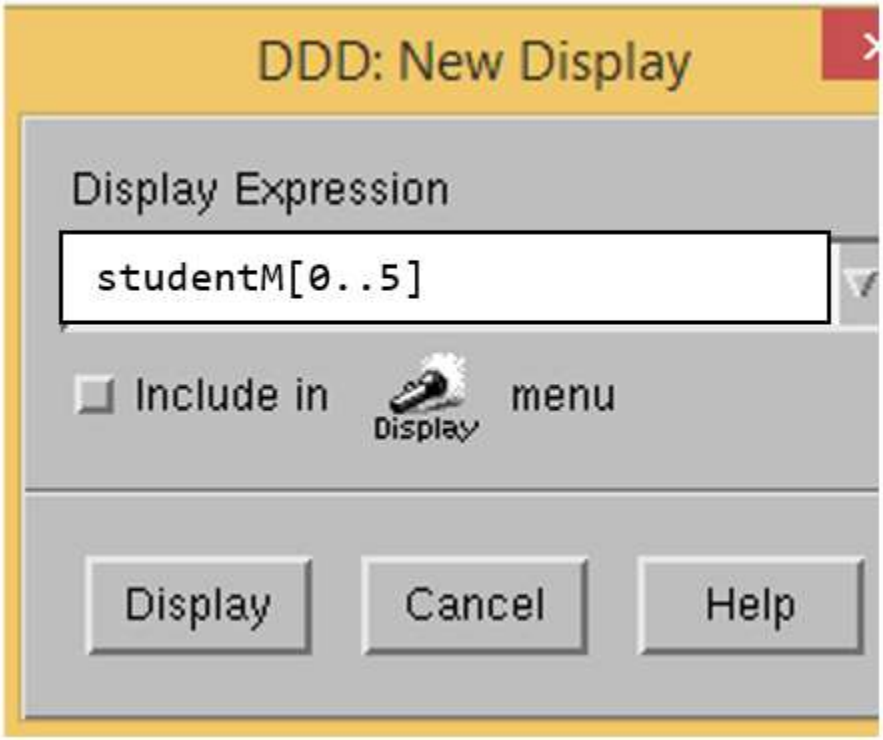


Figure 10: Showing the contents of studentM[1] using a New Display.

You can also display slices of an array. In the Display Area, right click and select **New Display**. In the Display Expression, specify a slice of an array as shown below:



When the array display appears, it may appear across the page. If your right click and select **Rotate,** it will display vertically.

How can I remove something from the Display Area?

Right click on it and select **Undisplay**.

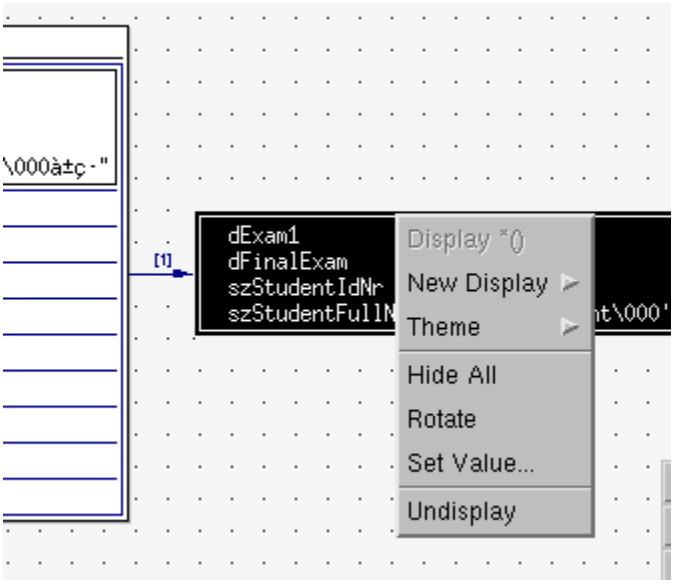
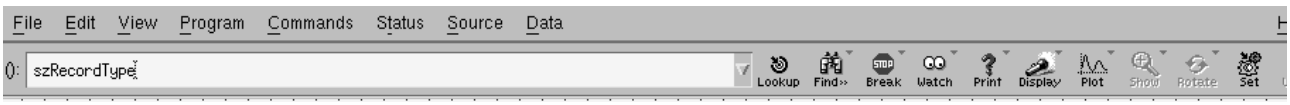


Figure 11: Removing a Display

How can I tell ddd to interrupt execution when a variable's value changes?

This **is** done by setting a **watch point. While running,** select what you want to watch using the mouse and press the **Watch** button (it looks like two eyeballs) in the Tool Bar. When you continue executing your code, it will interrupt execution at the point in the code where the value of that variable changes.



How can I fix the "GDB is Busy" error?

ddd is a GUI around the GDB debugger. Unfortunately, ddd sometimes gets hung without it being something you did incorrectly. To fix that problem:

- exit out of ddd completely
- Execute the following command in a Linux terminal session:

```
rm -r ~/.ddd
```