陈皓专栏 【空谷幽兰，心如皓月】

芝兰生于深谷，不以无人而不芳；君子修道立德，不为困穷而改节。

用GDB调试程序（六）

分类： 编程工具 2003-07-12 22:24 9719人阅读 评论(0) 收藏 举报

七、设置显示选项

GDB中关于显示的选项比较多，这里我只例举大多数常用的选项。

```
set print address
set print address on
```

打开地址输出，当程序显示函数信息时，GDB会显出函数的参数地址。系统默认为打开的，如：

```
(gdb) f
#0  set_quotes (lq=0x34c78 "<<", rq=0x34c88 ">>")
    at input.c:530
530         if (lquote != def_lquote)
```

```
set print address off
```

关闭函数的参数地址显示，如：

```
(gdb) set print addr off
(gdb) f
#0  set_quotes (lq="<<", rq=">>") at input.c:530
530         if (lquote != def_lquote)
```

```
show print address
```

查看当前地址显示选项是否打开。

```
set print array
set print array on
```

打开数组显示，打开后当数组显示时，每个元素占一行，如果不打开的话，每个元素则以逗号分隔。这个选项默认是关闭的。与之相关的两个命令如下，我就不再多说了。

```
set print array off
show print array
```

```
set print elements <number-of-elements>
```

这个选项主要是设置数组的，如果你的数组太大了，那么就可以指定一个<number-of-elements>来指定数据显示的最大长度，当到达这个长度时，GDB就不再往下显示了。如果设置为0，则表示不限制。

```
show print elements
```

查看print elements的选项信息。

(62817)
再谈“我是怎么招聘程序员
(54256)
一些重要的算法
(51996)
C++ 对象的内存布局（上
(46037)
哥是玩程序的
(43844)
清华大学出版社“抄袭事件
(43597)
恐怖的C++语言
(39271)

评论排行

清华大学出版社“抄袭事件 (323)
再谈“我是怎么招聘程序员 (264)
“清华大学出版社抄袭事件 (233)
我是怎么招聘程序员的 (233)
C++ 虚函数表解析 (203)
6个变态的C语言写的Hel (201)
恐怖的C++语言 (198)
优秀程序员的十个习惯 (165)
Java构造时成员初始化的 (146)
惹恼程序员的十件事 (130)

推荐文章

最新评论

跟我一起写 Makefile（一）
wangjilebeishang: 我的神啊，可以，顶《《《《学习

五个方法成为更好的程序员
雨中风铃_: 受教！

C++ 虚函数表解析
Walkerinwind: 第一个例子结束实际运行"经"果如下或许应该是结果?

“抄袭事件”判决书
fengyishang: 很遗憾，时隔七年，我才看到七年前这一起关于著作权的纠纷；但很幸运，因为最后的结果正是我们想要的，虽然...

C++ 虚函数表解析
zhiweiarm: 受益匪浅，学习了

22条经典的编程引言
雨中风铃_: 太经典了！

C++ 对象的内存布局（下）
Huang850615: cout << " "; pFun(); //D::f();这个地方出错了，把它改成这样：cout...

五个方法成为更好的程序员
冰冻牙膏: 寻找不同观点，取长补短

跟我一起写 Makefile（一）
luomuxiaoxiao98: Mark！

其实Unix很简单
lovelock: @yhmhappy2006:呵呵，我真不知道你干了什么。。。。

```
set print null-stop <on/off>
    如果打开了这个选项，那么当显示字符串时，遇到结束符则停止显示。这个选项默认为off。

set print pretty on
    如果打开printf pretty这个选项，那么当GDB显示结构体时会比较漂亮。如：

        $1 = {
          next = 0x0,
          flags = {
            sweet = 1,
            sour = 1
          },
          meat = 0x54 "Pork"
        }

set print pretty off
    关闭printf pretty这个选项，GDB显示结构体时会如下显示：

        $1 = {next = 0x0, flags = {sweet = 1, sour = 1}, meat = 0x54 "Pork"}

show print pretty
    查看GDB是如何显示结构体的。


set print sevenbit-strings <on/off>
    设置字符显示，是否按“/nnn”的格式显示，如果打开，则字符串或字符数据按/nnn显示，
如“/065”。

show print sevenbit-strings
    查看字符显示开关是否打开。

set print union <on/off>
    设置显示结构体时，是否显式其内的联合体数据。例如有以下数据结构：

    typedef enum {Tree, Bug} Species;
    typedef enum {Big_tree, Acorn, Seedling} Tree_forms;
    typedef enum {Caterpillar, Cocoon, Butterfly}
                  Bug_forms;

    struct thing {
      Species it;
      union {
        Tree_forms tree;
        Bug_forms bug;
      } form;
    };

    struct thing foo = {Tree, {Acorn}};

    当打开这个开关时，执行 p foo 命令后，会如下显示：
        $1 = {it = Tree, form = {tree = Acorn, bug = Cocoon}}

    当关闭这个开关时，执行 p foo 命令后，会如下显示：
        $1 = {it = Tree, form = {...}}
```

`show print union`
查看联合体数据的显示方式

`set print object <on/off>`
在C++中，如果一个对象指针指向其派生类，如果打开这个选项，GDB会自动按照虚方法调用的规则显示输出，如果关闭这个选项的话，GDB就不管虚函数表了。这个选项默认是off。

`show print object`
查看对象选项的设置。

`set print static-members <on/off>`
这个选项表示，当显示一个C++对象中的内容是，是否显示其中的静态数据成员。默认是on。

`show print static-members`
查看静态数据成员选项设置。

`set print vtbl <on/off>`
当此选项打开时，GDB将用比较规整的格式来显示虚函数表时。其默认是关闭的。

`show print vtbl`
查看虚函数显示格式的选项。

八、历史记录

当你用GDB的print查看程序运行时的数据时，你每一个print都会被GDB记录下来。GDB会以$1, $2, $3 .....这样的方式为你每一个print命令编上号。于是，你可以使用这个编号访问以前的表达式，如$1。这个功能所带来的好处是，如果你先前输入了一个比较长的表达式，如果你还想查看这个表达式的值，你可以使用历史记录来访问，省去了重复输入。

九、GDB环境变量

你可以在GDB的调试环境中定义自己的变量，用来保存一些调试程序中的运行数据。要定义一个GDB的变量很简单只需。使用GDB的set命令。GDB的环境变量和UNIX一样，也是以$起头。如：

```
set $foo = *object_ptr
```

使用环境变量时，GDB会在你第一次使用时创建这个变量，而在以后的使用中，则直接对其赋值。环境变量没有类型，你可以给环境变量定义任一的类型。包括结构体和数组。

`show convenience`
该命令查看当前所设置的所有的环境变量。

这是一个比较强大的功能，环境变量和程序变量的交互使用，将使得程序调试更为灵活便捷。例如：

```
set $i = 0
print bar[$i++]->contents
```

于是，当你就不必，print bar[0]->contents, print bar[1]->contents地输入命令了。输入这样的命令后，只用敲回车，重复执行上一条语句，环境变量会自动累加，从而完成逐个输出的功能。

十、查看寄存器

要查看寄存器的值，很简单，可以使用如下命令：

```
info registers
```
查看寄存器的情况。（除了浮点寄存器）

```
info all-registers
```
查看所有寄存器的情况。（包括浮点寄存器）

```
info registers <regname ...>
```
查看所指定的寄存器的情况。

寄存器中放置了程序运行时的数据，比如程序当前运行的指令地址（ip），程序的当前堆栈地址（sp）等等。你同样可以使用print命令来访问寄存器的情况，只需要在寄存器名字前加一个$符号就可以了。如：p $eip。

<-上一页　下一页->

分享到：

上一篇：用GDB调试程序（七）
下一篇：用GDB调试程序（三）

查看评论

暂无评论

您还没有登录,请[登录]或[注册]

* 以上用户言论只代表其个人观点，不代表CSDN网站的观点或立场