

Friday is in the CS Lab!

Before class:

Create an Eclipse project for ChangeMaker
with a simple Makefile

Make sure you can build and run

Import CS300CodeExamples into Eclipse!

Make sure you can build and run

Stop by my office with questions.

Stack

The stack is a LIFO (Last-in First-out) data structure

The only data element that can be removed is the most recently added element

Stack ADT

Specification

Elements: Stack elements can be of any type, but we will assume StackElement

Structure: Any mechanism for determining the elements order of arrival into the stack

Stack ADT Continued

Domain: The number of stack elements is bounded. A stack is considered full if the upper-bound is reached. A stack with no elements is considered empty.

type Stack;

Operations: There are seven operations as follows:

Stack ADT Continued

function create (s: Stack, isCreated: boolean)

results: if s cannot be created, isCreated is false; otherwise, isCreated is true, the stack is created and is empty

function terminate (s: Stack)

results: stack s no longer exists

Stack ADT Continued

function isFull (s: Stack)

results: returns true if the stack is full; otherwise false is returned

function isEmpty (s: Stack)

results: returns true if the stack is empty; otherwise, false is returned

function push (s: Stack, e: StackElement)

requires: isFull (s) is false

results: element e is added to the stack as the most recent element

Stack ADT Continued

function pop (s: Stack, e: StackElement)

requires: isEmpty(s) is not false

results: The most recently added element is removed and assigned to e

function peek (s: Stack, e: StackElement)

requires: isEmpty(s) is not false

results: The most recently added element is assigned to e but not removed

Testing your Data Structure

- Your customer will abuse your data structure
- Your data structure should never crash the customer's code
 - code defensively
- Test each each function
 - test each functions *requires* statement
 - test boundary conditions (full/empty)
 - test bad input
 - test functions called in the wrong order

Setup 9/16/2011

- Open up CS300CodeExamples
- In the Makefile, remove **bin/defineVsConst** from TARGETS
- Build the project
- Open fileIO.c

C Topics: File I/O

```
#include <stdio.h>
#include <errno.h>
```

- fopen

- fgetc

- fscanff

- fprintff

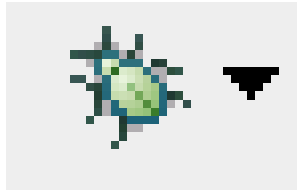
- close

- errno

- perror

```
FILE * fPtr = fopen("file.txt", "r");
char letter;
if( NULL == fPtr)
{
    perror("File did not open");
    return ERROR;
}
letter = fgetc(fPtr);
if( EOF != letter)
{
    printf("%c", letter);
}
fclose(fPtr);
```

Debugger



- Eclipse integrates a debugger just like Visual Studio
 - uses gdb

Open up CS300CodeExamples

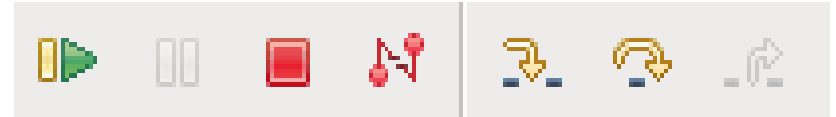
In the Makefile, remove bin/defineVsConst from TARGETS

- Build the project

- Open the Binaries list on the left
- Right-Click pointersWorksheet
- Debug As | Local C/C++ Application | gdb/mi

Debugger

The debugger stops on the first statement
in main()



Run to breakpoint

Pause

Terminate

Disconnect

Step Into

Step over

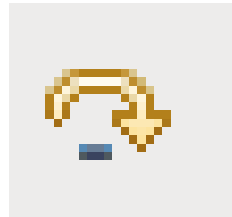
Step out of

Step one instruction

Notice the Variables in the top right

```
i  int  0
```

Press Step Over



Notice the Variables in the top right

```
i  int  1
```

Breakpoint

Right click the blue gutter beside printf()

Toggle Breakpoint

Run to Breakpoint



May need to
set Breakpoint Type
to C/C++

What is the value of i ?

Step Over

Check the console on the bottom

Conditional Breakpoint

Right Click that same breakpoint, the pale blue dot

Breakpoint Properties

Common

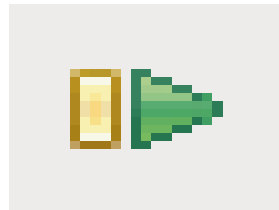
Condition: `i == 4`

Just C code!

Stop

Restart Debugging

Run to Breakpoint



Conditional Breakpoint

Be wary of function calls as conditions

Be wary of anything that accesses dynamic memory

(a null pointer in your condition!)

Error in testing breakpoint condition:
Cannot access memory at address 0x0

Ignore count: skip this break point X times

Actions: Sound/Log/Resume/External Tool

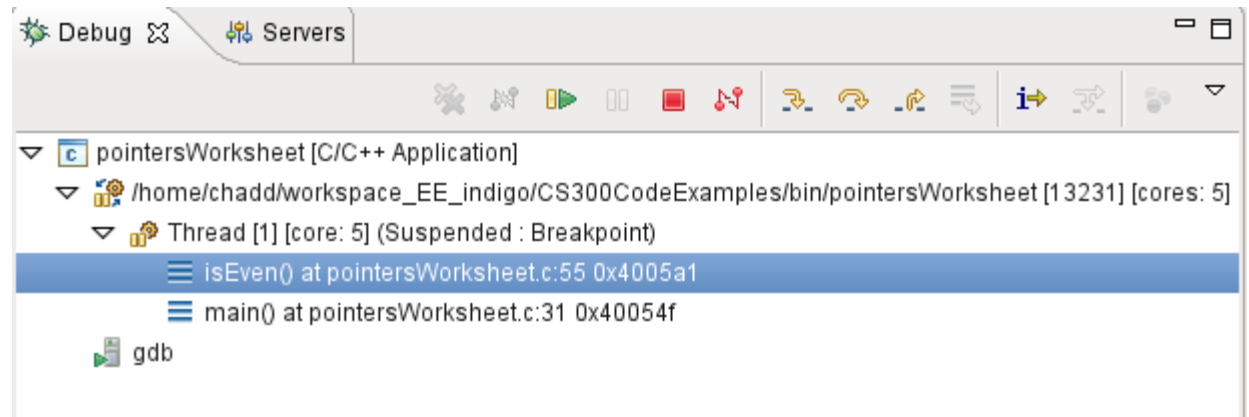
Filter: restrict to certain threads

Stack

Put a breakpoint on line 55 printf()

Disable breakpoint on line 33

Run



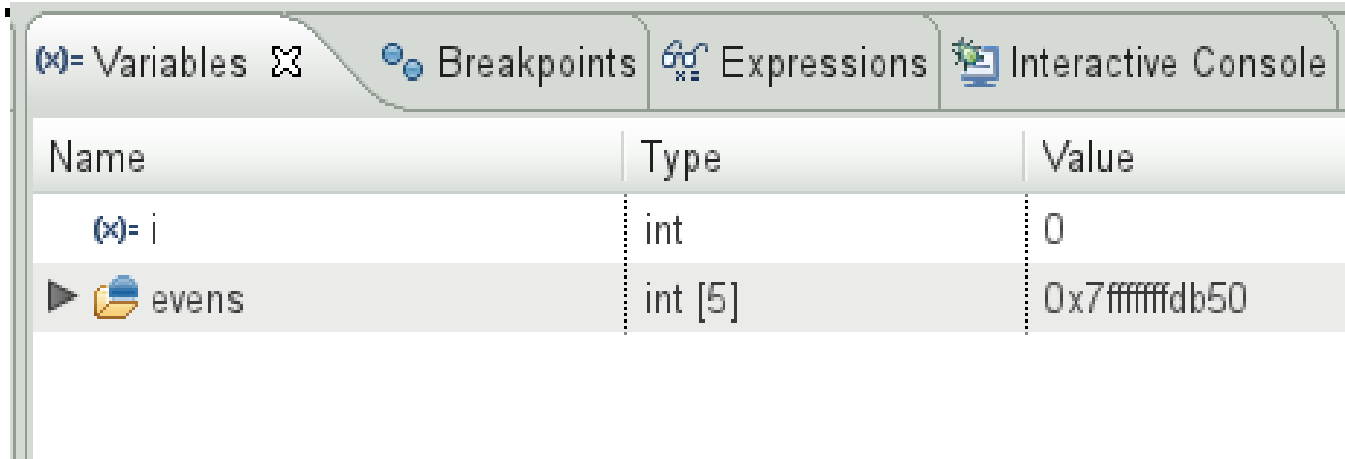
Select a function
to see variables in that function

Arrays

Add
to main().

```
int evens[MAX_NUMS];
```

Run



The screenshot shows a debugger's Variables view with the following data:

Name	Type	Value
(x)= i	int	0
▶ evens	int [5]	0x7fffffffdb50

Drop down evens in Variables view

charArraysAndStrings

charArraysAndStrings | Debug As

Flip back to
C/C++
Perspective

What is the first statement of main()

What is currently in charArray?

Step to the first printString()

Drop down pString & charArray

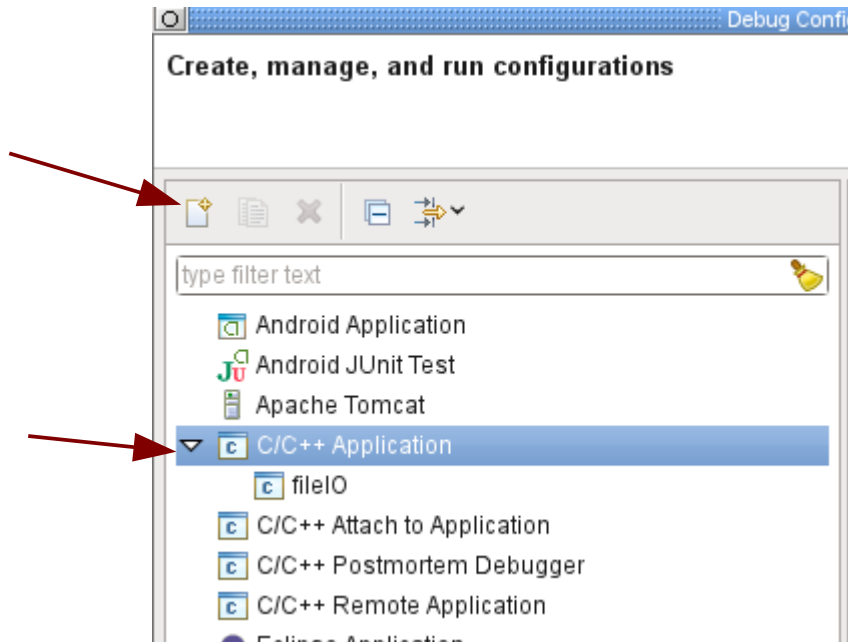
Right click pString | Display As Array

0 12 what is in pString[11]?

Right click pString | Restore Original Type

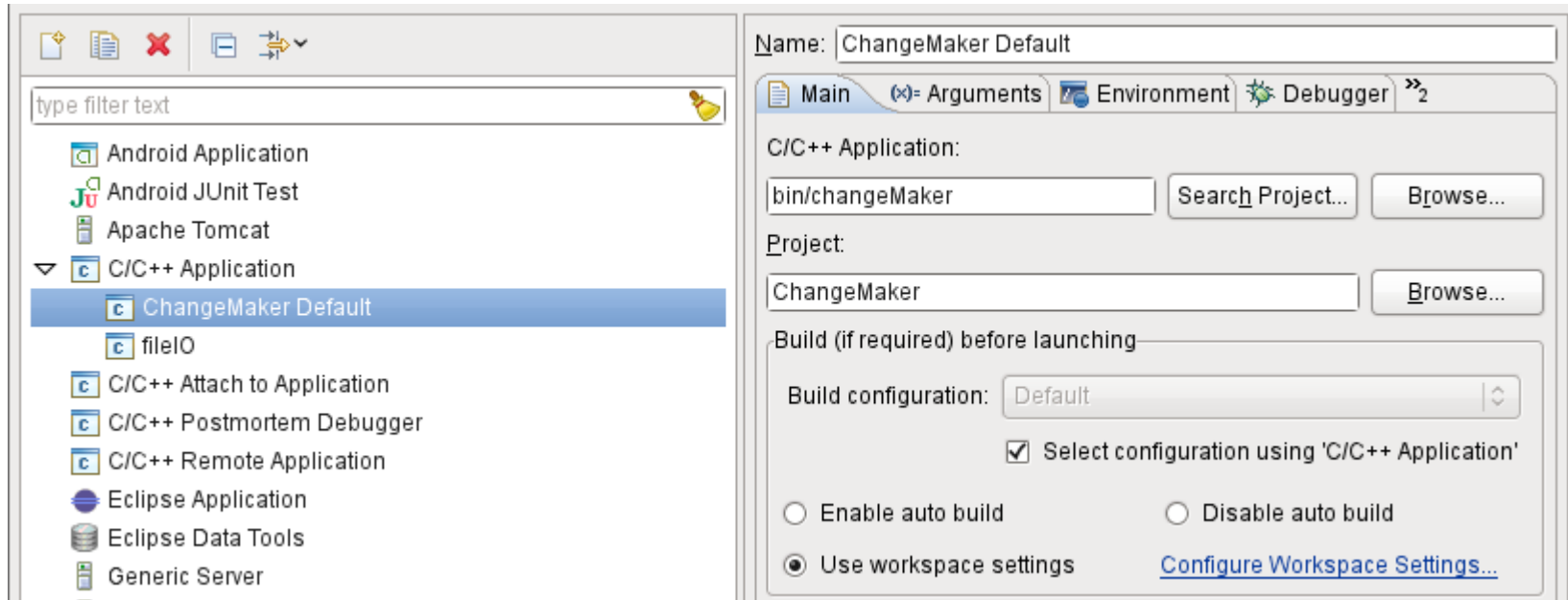
Debugger

Select C/C++ Application from the list
Press the New button



If there is only one executable in the project the fields will be automatically filled out

Debugger



Press Debug at the bottom to run the debugger

Debugger

- OR
 - Right click Project
- Debug As | Local C/C++ Application
- In the future, the configuration will show up in the bug drop down list

