Evolution of Programming Abstraction
Mechanisms: C-style Stack Implementations

Douglas C. Schmidt
d.schmidt@vanderbilt.edu
www.dre.vanderbilt.edu/~schmidt

Professor of Computer Science
Institute for Software Integrated Systems
Vanderbilt University
Nashville, Tennessee, USA
Learning Objectives in This Part of the Lesson

- The rest of this overview examines several alternative methods of implementing a Stack
- We’ll begin with C & evolve up to various C++ implementations

See en.wikipedia.org/wiki/Stack_(abstract_data_type)
C-style Stack Implementations
“Bare-Bones” C Stack Example

• First, consider the “bare-bones” C implementation:

```c
// Type of elements in the stack.
typedef int T;

// Max size of the stack.
#define MAX_STACK 10

int main() {
    T stack[MAX_STACK];
    int top;

    for (top = 0; top < MAX_STACK; ++top)
        stack[top] = top; // push

    while (--top >= 0)
        printf("top item = %d\n", stack[top]); // pop

    return 0;
}
```

See [CPlusPlus/tree/master/overview/capabilities/0-C-bare-bones](CPlusPlus/tree/master/overview/capabilities/0-C-bare-bones)
Pros of “Bare-Bones” C Stack Example

• Highly “efficient,” i.e., no function call overhead!
Cons of “Bare-Bones” C Stack Example

- It’s not very abstract, so small mistakes can cause big problems!