Data Abstraction Implementation in C

- Define the interface & implementation to a stack ADT in C

```c
int main() {
    /* Multiple stacks! */
    Stack s1, s2, s3;

    T item = 0;

    /* stack_create(&s1, MAX_STACK); */
    /* Oops, forgot to call stack_create()! */

    while (!stack_full(&s1))
        stack_push(&s1, item++);

    while (!stack_empty(&s1)) {
        T i;
        stack_top (&s1, &i);
        printf("top item = %d\n", i);
        stack_pop (&s1);
    }
}
```

See [CPlusPlus/tree/master/overview/capabilities/2-C-data-abstraction](CPlusPlus/tree/master/overview/capabilities/2-C-data-abstraction)
Pros of Data Abstraction Implementation in C

• Can support more than one stack
Cons of Data Abstraction Implementation in C

- No guaranteed initialization, termination, or assignment
- Still only one type of stack supported
- Too much overhead due to function calls
- No generalized error handling...
- The C compiler does not enforce information hiding, e.g.,

```c
s1.top_ = s2.stack_[0]; /* Violate abstraction */
s2.size_ = s3.top_; /* Violate abstraction */
```
End of C-style Stack Implementations