Abstract Base Class Example

The following example shows a use of multiple inheritance to create a new abstraction (i.e., a bounded stack) from two existing classes.

1. The first class (class Stack) provides an abstract interface that is filled in later on.

2. The second class (class Vector) provides the storage used to hold the stack elements.

**Advantage:**
- A very structured way to reuse existing code.

**Disadvantage:**
- All virtual function calls.
- Requires some advanced planning...

---

Abstract Base Class Example
(cont’d)

```cpp
#include <iostream>
#include <assert.h>
typedef int ELEMENT_TYPE;

class Stack {
public:
    // Note, no representation...
    virtual int nb_elements (void) = 0;
    virtual int empty (void) { return nb_elements () == 0; }
    virtual int full (void) = 0;
    virtual void push (ELEMENT_TYPE new_elem) = 0;
    virtual ELEMENT_TYPE top (void) = 0;
    virtual ELEMENT_TYPE pop (void) = 0;
    virtual void change_top (ELEMENT_TYPE ntop) {
        // Operations not known at this point...!
        this->pop (); this->push (ntop);
    }
    virtual void wipe_out (void) = 0;
    virtual ~Stack (void) { /* do nothing */ }
};
```

---

Abstract Base Class Example
(cont’d)

```cpp
class Vector {
private:
    int sz;
    ELEMENT_TYPE *buffer;
public:
    Vector (int s):
        sz (s), buffer (new ELEMENT_TYPE[s])
    {}
    virtual ELEMENT_TYPE &operator [] (int i) {
        return this->buffer[i];
    }
    virtual int size (void) { return this->sz; }
    virtual ~Vector (void) { delete this->buffer; }
};
```

---

Abstract Base Class Example
(cont’d)

```cpp
class Bounded_Stack : public Stack, private Vector {
private:
    int stack_top;
public:
    Bounded_Stack (int sz): Vector (sz), stack_top (0) {}
    virtual int nb_elements (void) {
        return this->stack_top;
    }
    virtual int full (void) {
        this->size () == this->nb_elements ();
    }
    virtual void push (ELEMENT_TYPE new_element) {
        (*this)[this->stack_top++] = new_element;
        // Vector::operator[] (this->stack_top++); };
```
Abstract Base Class Example
(cont'd)

- e.g.,

```cpp
#include <stream.h>
#include "bounded_stack.h"
int main (int argc, char *argv[]) {
    if (argc != 2) {
        cerr << "usage: " << argv[0] << " numeric-arg\n";
        exit (1);
    }
    int size = atoi (argv[1]);
    srand (time (0));

    Bounded_Seq bounded_stack (size);

    while (! bounded_stack.full ())
        bounded_stack.push (random ());

    while (! bounded_stack.empty ())
        cout << bounded_stack.pop () << "\n";
    // call destructor
}
```