```
class Pila
{
public:
     Pila(size_t depth);
     Pila(const Pila & r);
     void operator= (const Pila & rhs);

     void Push (std::string s);
     std::string Pop ();
     std::string Peek ();
     bool IsEmpty ();
     bool IsFull ();
     void Clear ();

     size_t GetDepth ();
     size_t GetTop ();

private:
     size_t depth;
     // depth no tiene valor predeterminado

     std::vector<std::string> stack;
     // std::vector será la estructura subyacente que soportará
     // la implementación de la pila

     size_t top {0};
     // el índice de la pila siempre empieza en cero
};
```

```
Pila::Pila (size_t _depth) : depth {_depth}, stack {_depth}
{
}

size_t Pila::GetTop ()
{
     return this->top;
}

size_t Pila::GetDepth ()
{
     return this->depth;
}

bool Pila::IsEmpty()
{
     // por hacer
}

bool Pila::IsFull()
{
     // por hacer
}

void Pila::Push(std::string s)
{
     // por hacer
}

std::string Pila::Pop ()
{
     // por hacer
}

std::string Pila::Peek()
{
     // por hacer
}

void  Pila::Clear()
{
     // por hacer
}
```

```
/*--------------------------------------------------------------------
*                          Driver program
*-------------------------------------------------------------------*/
int main(void)
{
     Pila p {5};

     std::cout << ".depth = " << p.GetDepth () << std::endl;
     std::cout << ".top   = " << p.GetTop () << std::endl;

     for (size_t i {0}; p.IsFull () == false; ++i) {
          std::string s;

          std::cout << i << ": ";
          std::cin >> s;

          p.Push (s);
          std::cout << ".top   = " << p.GetTop () << std::endl;
     }

     std::cout << "=========================" << std::endl;

     while (p.IsEmpty () == false) {

          std::cout << p.Pop () << std::endl;
          std::cout << ".top   = " << p.GetTop () << std::endl;
     }

     return 0;
}
```