```cpp
#include <iostream>
using  namespace  std  ;

struct  elemenet
{
    int                data  ;
    struct elemenet *  next  ;
};

typedef  struct elemenet Element  ;
typedef  Element *       Stack    ;

Stack  create_stack()
{
    return  NULL  ;
}

bool  empty_stack(Stack stk)
{
    if(stk == NULL)
        return  true  ;
    else
        return  false ;
}

int  deapth_stack(Stack stk)
{
    Stack  ptr    =  stk  ;
    int    count  =  0    ;

    while(ptr != NULL)
    {
        count++  ;
        ptr  =  ptr->next  ;
    }

    return  count  ;
}

Stack  push_stack(Stack stk, int val)
{
    Stack  ptr  =  new  Element  ;
    ptr->data   =  val  ;

    ptr->next  =  stk  ;
    stk        =  ptr  ;

    return  stk  ;
}

Stack  pop_stack(Stack stk, int &val)
{
    if(!empty_stack(stk))
    {
        Stack  ptr  =  stk  ;

        val  =  stk->data  ;
        stk  =  stk->next  ;
        delete  ptr        ;

        return  stk  ;
    }
    else
    {
        cout  <<  "Error, the stack is already empty."  <<  endl  ;
        return  NULL  ;
    }
}

int  top_stack(Stack stk)
{
    if(empty_stack(stk))
    {
        cout  <<  "Error, Stack is empty."  <<  endl  ;
        return  0  ;
    }

```

```
 78      return  stk->data  ;
 79  }
 80
 81  Stack  replace_top_stack(Stack stk, int val)
 82  {
 83      if(empty_stack(stk))
 84      {
 85          cout  <<  "Error, Stack is empty."  <<  endl  ;
 86          return  NULL  ;
 87      }
 88
 89      stk->data  =  val  ;
 90
 91      return  stk  ;
 92  }
 93
 94  void  display_stack(Stack stk)
 95  {
 96      Stack  ptr    =  stk  ;
 97
 98      if(empty_stack(stk))
 99      {
100          cout  <<  "Stack is empty."  <<  endl  ;
101          return  ;
102      }
103
104      while(ptr != NULL)
105      {
106          cout  <<  ptr->data  <<  " "  ;
107          ptr  =  ptr->next  ;
108      }
109      cout  <<  endl  ;
110
111      return  ;
112  }
113
114  int  main()
115  {
116      Stack  stk  =  create_stack()  ;
117
118      display_stack(stk)  ;
119
120      stk  =  push_stack(stk, 1)  ;
121      stk  =  push_stack(stk, 2)  ;
122      stk  =  push_stack(stk, 3)  ;
123
124      display_stack(stk)  ;
125
126      cout  <<  "The top of the stack is  : "  <<  top_stack(stk)  <<  endl  ;
127
128      stk  =  replace_top_stack(stk, 4)  ;
129
130      display_stack(stk)  ;
131
132      cout  <<  "The depth of the stack is  : "  <<  deapth_stack(stk)  <<  endl  ;
133
134      int  val  ;
135
136      stk  =  pop_stack(stk, val)  ;
137      cout  <<  "The value popped from the stack is  : "  <<  val  <<  endl  ;
138      display_stack(stk)  ;
139
140      stk  =  pop_stack(stk, val)  ;
141      cout  <<  "The value popped from the stack is  : "  <<  val  <<  endl  ;
142      display_stack(stk)  ;
143
144      stk  =  pop_stack(stk, val)  ;
145      cout  <<  "The value popped from the stack is  : "  <<  val  <<  endl  ;
146      display_stack(stk)  ;
147
148      return  0  ;
149  }
```