

## Devoir Final POO C++

**Préparé par :**  
**Chaimae Bouassab**  
**Licence en :**  
**Ingénierie du Développement**  
**d'Applications Informatiques**

```
#include <iostream>

class Stack {
private:
    struct Node {
        int data; // data represente l'entier n.
        Node* previous;
    };

    Node* sommet; // Elle pointe vers le dernier élément ajouté à la pile.

public:
    Stack(int x1 = 20) {
        data = x1;
    }
    Stack() : sommet(nullptr) {}

    void operator<<(int data) {
        Node* element = new Node{data, sommet};
        sommet = element;
    }

    int pile_pop(int &data) {
        if (sommet == nullptr) {
            return -1; // La pile va retourner une erreur.
        }

        Node* supp_element = sommet;
        data = supp_element->data;
        sommet = sommet->previous;
        delete supp_element;
    }
};
```

```

        return 0;
    }

    void affiche() {
        Node* courant = sommet;
        while (courant != nullptr) {
            std::cout << "la valeur : " << courant->data << std::endl;
            courant = courant->previous;
        }
    }

    void pile_clear() {
        int data;
        while (pile_pop(data) == 0) {
            // Continue à dépiler jusqu'à ce que la pile soit vide.
        }
    };

int main() {
    Stack stack;
    stack.operator<<(12);
    stack.operator<<(1);
    stack.operator<<(2);
    stack.operator<<(3);

    std::cout << "Contenu de la pile : " << std::endl;
    stack.affiche();

    int popped_data;
    int resultat = stack.pile_pop(popped_data);
    if (resultat == 0) {
        std::cout << "Donnée que on va retirer : " << popped_data << std::endl;
    } else {
        std::cout << "La pile est vide." << std::endl;
    }

    return 0;
}

```

```

1  #include <iostream>
2
3  class Stack {
4  private:
5      struct Node {
6          int data;
7          Node* previous;
8      };
9
10     Node* sommet;
11     // Elle pointe vers le dernier
12     // élément ajouté à la pile.
13 public:
14     Stack(int x1 = 20) {
15         data = x1;
16     }
17     Stack() : sommet(nullptr) {}
18
19     void operator<<(int data) {
20         Node* element = new
21         Node(data, sommet);
22         sommet = element;
23     }
24
25     int pile_pop(int &data) {
26         if (sommet == nullptr)
27         {
28             return -1;
29             // La pile va retourner une er
30             // reur.
31         }
32         Node* supp_element =
33         sommet;
34         data = supp_element->
35         data;
36         sommet = sommet->
37         previous;
38         delete supp_element;
39         return 0;
40     }
41
42     void affiche() {
43         Node* courant = sommet;
44         while (courant !=
45         nullptr) {
46             std::cout <<
47             "la valeur : " << courant->
48             data << std::endl;
49             courant = courant->
50             previous;
51         }
52     }
53 };
54
55 int main() {
56     Stack stack;
57     stack.operator<<((12));
58     stack.operator<<((1));
59     stack.operator<<((2));
60     stack.operator<<((3));
61
62     std::cout <<
63     "Contenu de la pile : " << std
64     ::endl;
65     stack.affiche();
66
67     int popped_data;
68     int resultat = stack.
69     pile_pop(popped_data);
70     if (resultat == 0) {
71         std::cout <<
72         "Donnée que on va retirer : "
73         << popped_data << std::endl;
74     } else {
75         std::cout <<
76         "La pile est vide." << std::
77         endl;
78     }
79     return 0;
80 }

```