C++: יוצרים, והורסים

1

המחלקה Stack
תזכורת

```
class Stack {
private:
        int* array;
        int size, top_index;
public:
        Result init (int size) ;
        void destroy() ;
        Result push (int e);
        Result pop ();
        Result top(int& e);
        Result print() ;
} ;
```

ומה יקרה אם המשתמש ישכח לקרוא ל-init
או ל-destroy ???

2

יוצרים והורסים

הבעיה - בהגדרת הטיפוס אנו משאירים **פתח לטעויות** של המשתמש היכול לשכוח לאתחל אובייקטים לפני השימוש בהם (לא לקרוא ל init) דבר העלול לגרום לכך שהתוכנית תתנהג באופן בלתי צפוי, או לשכוח להרוס אובייקט וכך לגרום לדליפת זיכרון.

```
class Stack {
 private:
  int* array;
  int size, top_index ;
 public:
  Stack (int size) ;
  ~Stack() ;
 ...
```

יתר על כן, פעמים רבות נוצרים **אובייקטים זמנים** (למשל, בהעברת והחזרת פרמטרים מפונקציות) שאין למשתמש יכולות לאתחלם בצורה מפורשת.

הפתרון – פונקציות אתחול והריסה שנקראות אוטומטית כאשר האובייקט נוצר וכאשר הוא אמור להיהרס.

מעתה, בכל פעם שנגדיר משתנה מטיפוס Stack נדרש להעביר ליוצר את הפרמטרים הדרושים לצורך אתחול האובייקט, ובכל פעם שהאובייקט לא יהיה נגיש יותר הקומפיילר יקרא להורס אשר ישחרר את האובייקט.

שמות היוצרים (constructors) כשם המחלקה. שם ההורס (destructor) כשם המחלקה שלפניה ~.

3

מימוש Constructors ו Destructors

שימו לב C'tor ו D'tor לא מחזירים ערכים !

```
Stack::Stack (int s) {
        array = new int[s] ; // חסרה בדיקה
        top_index = 0; size = s ;
}
Stack::~Stack() {
        delete[] array;
}
```

<u>הערה:</u>

כמעט לכל מחלקה נדרש להגדיר יוצרים לאתחול שדותיה.

הגדרת הורסים תתבצע בד"כ עבור מחלקות המקצות זיכרון דינמי בעצמן. במקרה זה על ההורס לשחררו.

אם איננו מגדירים יוצרים והורסים, מוגדרים אוטומטית יוצרים והורסים דיפולטיים.

4

שימוש במחסנית שיש לה יוצרים והורסים

```
#include "Stack.h"
int main() {
    int  i;
    Stack s(100) ;                    // the c'tor is called with size 100
    Stack* ps = new Stack(100);  // same here
    if (ps == NULL) exit(1);
    s.push (1); s.push(213); s.pop (); s.top(i);
    ps->push(1); ps->push(2); ps->pop() ;
    delete ps ;                       //  the d'tor is called for ps
}
// the d'tor is called for s
```

5

זמני קריאה של יוצרים והורסים

- קיימות 4 דרכים להקצאת משתנים. זמני הקריאה של היוצרים וההורסים תלויים באופן שבו הוקצה האובייקט.

משתנים לוקאליים

היוצר נקרא בכל פעם שהתוכנית מגיעה להכרזת המשתנה.
ההורס בכל פעם שהתוכנית יוצאת מהתחום בו הוגדר המשתנה

משתנים גלובאליים

היוצר נקרא עם תחילת התוכנית (לפני ה-main).
ההורס עם סיום התוכנית (לאחר סיום main).

משתנים דינמיים

היוצר נקרא בכל פעם שמוקצה אובייקט ע"י **new**.
ההורס בכל פעם שאובייקט משוחרר ע"י **delete**.

משתנים סטאטיים

היוצר נקרא בפעם הראשונה שהתוכנית מגיע לתחום בו מוגדר המשתנה.
ההורס עם סיום התוכנית.

דוגמא לזמני קריאה

```
#include "Stack.h"
Stack glbals(100);                    // globals c'tor is called
int main() {
    Stack locals(50) ;                // locals c'tor is called
    Stack* ps = new Stack(600);       // ps c'tor is called
    Stack* ps2 = new Stack(600);      // ps2 c'tor is called
    delete ps ;          // ps destructor is called
    return 0;                 // locals destructor is called
}
                         // globals destructor is called
                         // ps2 destructor is never called !
```

7

Advanced C'tors and D'tors

```
class TwoStack .. {
    Stack s1, s2 ;
public :
    TwoStack(int s) ;
    ..
};

TwoStack::TwoStack (int size) :
    s1(100), s2(size * 5) {
    …
}
```

- בכדי להקצות **מערכים** של אובייקטים ממחלקה כלשהי, לאותה מחלקה נדרש שיהיה יוצר שלא מקבל פרמטרים (או שיש לו ערכים דיפולטיים לכל הפרמטרים שלו כך שאין צורך לציין פרמטרים במפורש).
- ניתן להגדיר מחלקות אשר יש בהן שדות שהם עצמם עצמים של מחלקות אחרות. במקרה זה:
  - כאשר נוצר אובייקט של המחלקה, ראשית מאותחלים כל שדותיו. כאשר נהרס אובייקט כזה, ראשית נקרא ההורס שלו ורק אח"כ של שדותיו.
  - אם יוצרי השדות זקוקים לפרמטרים ניתן לבצע זאת ע"י רשימת אתחול.

8

רשימות אתחול

- הדרך המקובלת לאתחול שדות פנימיים של מחלקה.
- מייד אחרי הצהרת היוצר ולפני גוף היוצר (החלק שבתוך ה-{}) מופיעות נקודותיים ואז רשימה של השדות הפנימיים, כשהם מופרדים על ידי פסיקים.
- כל מופע של שדה ברשימת האתחול הוא למעשה קריאה לאחד מהיוצרים של השדה.
- לכל שדה כותבים בסוגריים את הערכים שמעבירים ליוצר שלו (ויכולים להיות תלויים בפרמטרים שהועברו ליוצר הראשי).
- סדר הפעלת היוצרים **אינו** הסדר בו הם מופיעים ברשימת האתחול אלא בו השדות מופיעים בהגדרת המחלקה.
- רשימת אתחול כדאית על פני "אתחול" בתוך הפונקציה כיוון שחוסכים אתחול ראשוני בזבל.

9