

פקולטה: מדעי הטבע

מחלקה: מדעי המחשב

שם הקורס: מבנה זיכרון ושפת ++C

קוד הקורס: 2-7027810 כל הקבוצות - קבוצות 1,2,3,4

מועד \_\_א\_\_\_ סמסטר \_\_\_ב\_\_\_תאריך בחינה:\_\_4/7/2018\_\_\_ **פתרון**

משך הבחינה: 3 שעות

שם המרצים: אראל סגל-הלוי, חרות סטרמן

## **שאלה** **1 [10 נק']**

נתונה התוכנית:

01 #include <iostream>

02 using namespace std;

03

04 class triplet {

05     double x,y,z;

06 public:

07     triplet(double x, double y,double z): x(x), y(y), z(z) {}

08     friend ostream& operator<< (ostream& out, const triplet& t);

09 };

10

11 ostream& operator<< (ostream& out, const triplet& t) {

12     out << "(" << t.x << "," << t.y << "," << t.z << ")";

13 }

14

15 int main() {

16     triplet t(1,2,3);

17     cout << t << endl;

18 }

כשמריצים make, מתקבל הפלט הבא:

main.cpp:13:1: warning: control reaches end of non-void function [-Wreturn-type]

}

^

1 warning generated.

./a.out

make: \*\*\* [all] Illegal instruction (core dumped)

א [5 נק']. מה סוג השגיאה (קומפילציה / קישור / זמן-ריצה / לוגית)? מה בדיוק גורם לשגיאה?

*שגיאת זמן ריצה.*

*באופרטור הפלט (שורות 11-13) אין פקודת return בסוף.*

*כתוצאה מכך, האופרטור מחזיר זבל - רפרנס למקום לא חוקי.  
בשורה 17, הפעולה cout << t מחזירה רפרנס-זבל.*

*מייד לאחר מכן מנסים לכתוב לתוכו endl - הכתיבה נעשית לכתובת לא חוקית ולכן נזרקת חריגה על הוראה לא חוקית.*

***הערה****: השגיאה לא בהכרח מתגלה בכל מצב. למעשה, כשפונקציה לא מחזירה ערך, ההתנהגות של התוכנית היא "לא מוגדרת" – undefined. זה אומר שהיא יכולה להחזיר חריגה, אבל היא יכולה גם לעבוד בצורה תקינה או לא תקינה. זה כמו מצביע לא מאותחל: ייתכן שיש בו ערך לא חוקי שיגרום לחריגה, אבל ייתכן שבמקרה יש בו ערך חוקי שלא יגרום לשום חריגה.*

ב [5 נק']. יש לתקן את הבאג ע"י שינוי/הוספה של שורה אחת. איזה שינוי יש לבצע כך שהתוכנית תרוץ בלי שגיאות ותדפיס (1,2,3)?

*אחרי שורה 12 ולפני שורה 13, להוסיף return out.*

## **שאלה 2 [10 נק']**

נתון הקוד החלקי הבא (שלוש נקודות מציינות קוד שהושמט כי אינו רלבנטי לשאלה):

01 #include <iostream>

02 using namespace std;

03

04 class LargeMatrix {

05 public: // ...

06 LargeMatrix(int width, int height) {

07 cout << "standard ctor" << endl; // ...

08 }

09 LargeMatrix(const LargeMatrix& other) {

10 cout << "copy ctor" << endl; // ...

11 }

12 };

13

14 int sum\_of\_values(LargeMatrix m) {

15 cout << "sum\_of\_values" << endl;

16 int result = 0;

17 // do calculation ...

18 return result;

19 }

20

21 int main() {

22 LargeMatrix m1(1000,1000);

23     // ...

24  cout << sum\_of\_values(m1) << endl;

25 }

התוכנית מתקמפלת ורצה, אבל יש בה באג הגורם לכך שהתוכנית רצה מאד לאט.

הפלט המתקבל כולל את ארבע השורות הבאות:

standard ctor

copy ctor

sum\_of\_values

0

א [5 נק']. מהו הבאג? (ציינו מספרי שורות).

*בשורה 14, המשתנה m מועבר by value.*

*כתוצאה מכך, כשקוראים לפונקציה sum\_of\_values, נקרא הבנאי המעתיק.*

*הבנאי המעתיק של מטריצה גדולה, כנראה, מעתיק את כל המטריצה וזה לוקח הרבה זמן.*

ב [5 נק']. יש לתקן את הבאג ע"י שינוי/הוספה של שורה אחת. איזה שינוי יש לבצע כך שהתוכנית תרוץ מהר ובלי שגיאות?

*בשורה 14, להעביר את המשתנה m כרפרנס:* ***LargeMatrix& m****.*

## **שאלה 3 [10 נק']**

נתונה התוכנית הבאה:

class XX {

    int ixx;

};

class YY: public XX {

    int iyy;

};

class ZZ {

    YY oyy;

    int izz;

};

int main() {

    ZZ ozz;

}

התוכנית בשפת ++C. התוכנית מתקמפלת ורצה בלי שגיאות.

א. באיזה איזור בזיכרון נמצא המשתנה ixx (**מחסנית** / ערימה / אחר)?

ב. באיזה איזור בזיכרון נמצא המשתנה iyy (**מחסנית** / ערימה / אחר)?

ג. באיזה איזור בזיכרון נמצא המשתנה izz (**מחסנית** / ערימה / אחר)?

ד. התוכנית תורגמה לשפת Java. התוכנית הראשית נשארה ללא שינוי:

 public static void main(String[] args) {

        ZZ ozz;

 }

התוכנית מתקמפלת ורצה בלי שגיאות.

באיזה איזור בזיכרון נמצא המשתנה ixx (מחסנית / ערימה / **אחר**)?

*[לא נמצא בשום מקום כי העצם לא נוצר - זה רק פוינטר].*

ה. התוכנית תורגמה לשפת Java. התוכנית הראשית שונתה ל:

 public static void main(String[] args) {

        ZZ ozz = new ZZ();

  }

התוכנית מתקמפלת ורצה בלי שגיאות.

באיזה איזור בזיכרון נמצא המשתנה ixx (מחסנית / ערימה / **אחר**)?

*[לא נמצא בשום מקום - העצם מסוג ZZ אמנם נוצר, אבל השדה מסוג YY לא נוצר - הוא רק פוינטר].*

## **שאלה 4 [10 נק']**

נתונה תוכנית המורכבת משלושה קבצים.

**AbstractShape.h:**

01

02 class AbstractShape {

03 public:

04     virtual void plot();

05 };

**Circle.h:**

11 #include "AbstractShape.h"

12 #include <iostream>

13 class Circle: public AbstractShape {

14 public:

15     void plot() { /\* code for plotting a circle. not shown. \*/ }

16 };

**main.cpp**:

21 #include "Circle.h"

22 int main() {

23     AbstractShape\* s = new Circle();

24 s->plot();

25 }

כשמריצים make מתקבלת הודעת השגיאה הבאה:

/tmp/main-3bcff2.o: In function `AbstractShape::AbstractShape()':

main.cpp:(.text.\_ZN13AbstractShapeC2Ev[\_ZN13AbstractShapeC2Ev]+0x6): undefined reference to `vtable for AbstractShape'

א [5 נק']. מה סוג השגיאה (קומפילציה / קישור / זמן-ריצה / לוגית)? מה בדיוק גורם לשגיאה?

*שגיאת קישור.*

*בקובץ AbstractShape.h מגדירים פונקציה וירטואלית בשם plot, אבל לא מממשים אותה בשום מקום.*

*כתוצאה מכך לא נוצרת טבלת פונקציות וירטואליות עבור AbstractShape.*

*הבנאי של AbstractShape צריך לאתחל את ה-vptr של המחלקה כך שיצביע לטבלה הוירטואלית, אבל הטבלה לא קיימת.*

*המקשר מגלה את זה ומתלונן על הפניה לא מוגדרת ל-vtable for AbstractShape.*

ב [5 נק']. יש לתקן את הבאג ע"י שינוי/הוספה של שורה אחת. איזה שינוי יש לבצע כך שהתוכנית תרוץ בלי שגיאות ותצייר עיגול?

*כיוון שהמחלקה AbstractShape מייצגת צורה מופשטת, לא אמור בכלל להיות לה בנאי. הדרך הטובה ביותר להשיג מטרה זו היא להפוך את הפונקציה plot לוירטואלית-טהורה ע"י הוספת "=0" בשורה 4:*

*04     virtual void plot()* ***= 0****;*

*פתרון אפשרי אבל פחות טוב הוא להוסיף מימוש ריק לפונקציה, למשל:*

*04     virtual void plot()* ***{}***

## שאלה 5 [10 נק']

נתונה התוכנית:

01 #include <set>

02 #include <iostream>

03 using namespace std;

04

05 struct XX {

06     int x;

07     XX(int x): x(x) {}

08 };

09

10 int main() { // the program should print "100"

11     multiset<XX> myset;

12     myset.emplace(1);

13     myset.emplace(0);

14     myset.emplace(0);

15     for (auto y: myset)

16         cout << y.x;

17 }

כשמריצים make מתקבל הפלט הבא:

/usr/bin/../lib/gcc/x86\_64-linux-gnu/7.1.0/../../../../include/c++/7.1.0/bits/stl\_function.h:386:20: error: invalid operands to binary expression ('const XX' and 'const XX')

{ return \_\_x < \_\_y; }

~~~ ^ ~~~

/usr/bin/../lib/gcc/x86\_64-linux-gnu/7.1.0/../../../../include/c++/7.1.0/bits/stl\_tree.h:2069:10: note: in instantiation of member function 'std::less<XX>::operator()' requested here

\_\_x = \_M\_impl.\_M\_key\_compare(\_\_k, \_S\_key(\_\_x)) ?

^

/usr/bin/../lib/gcc/x86\_64-linux-gnu/7.1.0/../../../../include/c++/7.1.0/bits/stl\_tree.h:2381:19: note: in instantiation of member function 'std::\_Rb\_tree<XX, XX, std::\_Identity<XX>, std::less<XX>, std::allocator<XX> >::\_M\_get\_insert\_equal\_pos' requested here

auto \_\_res = \_M\_get\_insert\_equal\_pos(\_S\_key(\_\_z));

^

/usr/bin/../lib/gcc/x86\_64-linux-gnu/7.1.0/../../../../include/c++/7.1.0/bits/stl\_multiset.h:448:16: note: in instantiation of function template specialization 'std::\_Rb\_tree<XX, XX, std::\_Identity<XX>, std::less<XX>, std::allocator<XX> >::\_M\_emplace\_equal<int>' requested here

{ return \_M\_t.\_M\_emplace\_equal(std::forward<\_Args>(\_\_args)...); }

^

main:12:8: note: in instantiation of function template specialization 'std::multiset<XX, std::less<XX>, std::allocator<XX> >::emplace<int>' requested here

myset.emplace(1);

א [5 נק']. מה סוג השגיאה (קומפילציה / קישור / זמן-ריצה / לוגית)? מה בדיוק גורם לשגיאה?

*שגיאת קומפילציה.*

*בשורה11 אנחנו יוצרים עצם מסוג multiset של XX.*

*הסוג multiset הוא בעצם תבנית (template) שבונה קבוצה מסודרת, ומשתמשת באופרטור "קטן מ" כדי לסדר.*

*אבל, לסוג XX אין אופרטור "קטן מ". השגיאה מתגלה כשהקומפיילר מקמפל את הקוד של התבנית - לכן השגיאה היא בקוד של הספריה התקנית (stl).*

ב [5 נק']. איך אפשר לתקן את התוכנית כך שתרוץ בלי שגיאות ותדפיס 100?

*אפשרות אחת היא להוסיף אופרטור "קטן מ" למחלקה XX. אבל כדי שהתוכנית תדפיס 100, האופרטור צריך לשים את 1 לפני 0. למשל, אפשר להוסיף בשורה 9:*

*bool operator< (const XX& a, const XX& b) { return a.x* ***>*** *b.x; }*

*פתרון נוסף: להוסיף אופרטור בתוך המחלקה - בין שורות 7,8:*

*bool operator< (const XX& other) const {return x > other.x;}*

## שאלה 6 [40 נק']

נתונה התוכנית:

01 #include <cmath>

02 #include <iostream>

03 #include <vector>

04 using namespace std;

05

06 class PositiveMatrix {

07     double\* vals;

08     int rows,cols;

09 public:

10     PositiveMatrix(int rows, int cols):

11     rows(rows), cols(cols), vals(new double[rows\*cols]) {

12         for (int i=0; i<rows\*cols; ++i)

13             vals[i]=0.0;

14     }

15

16     double operator() (int x, int y) const {

17         return vals[x\*cols+y];

18     }

19

20     double& operator() (int x, int y) {

21         return vals[x\*cols+y];

22     }

23 };

24

25 int main() {

26     PositiveMatrix m1(10,4);

27     cout << m1(1,2) << endl; // prints 0 (OK)

28     m1(1,2) = 3.0;

29     cout << m1(1,2) << endl; // prints 3 (OK)

30

31     PositiveMatrix m2{m1};

32     m2(1,2) = 5;

33     cout << m1(1,2) << endl; // prints 5 (should print 3)

34

35 m1 = m2;

36 m1(1,2) = 7;

37     cout << m2(1,2) << endl; // prints 7 (should print 5)

38

39     m1(1,2) = -5.0; // should throw an exception "illegal value"

40 }

התוכנית מתקמפלת ורצה אולם המחלקה PositiveMatrix לא גמורה - יש להשלים בה מספר חלקים לפי הפירוט בסעיפים הבאים. **אין לשנות את התוכנית הראשית** - עליכם לוודא שהתוכנית הראשית כפי שהיא כתובה עכשיו רצה בלי שגיאות. אם השינוי שלכם דורש להחליף קוד שכבר קיים במחלקה PositiveMatrix - ציינו בפירוש את מספרי השורות שבהן יש להחליף את הקוד ומה יש לכתוב במקומו.

א [10 נקודות]. יש להוסיף קוד למחלקה PositivcMatrix כך שלא תהיה דליפת זיכרון בתוכנית.

*להוסיף מפרק:*

*~PositiveMatrix()*

*{*

*delete[] vals;*

*}*

ב [10 נקודות]. יש להוסיף קוד למחלקה PositivcMatrix כך שבשורה 33 יודפס הערך הנכון - 3.

*להוסיף בנאי מעתיק:*

*PositiveMatrix(const PositiveMatrix& other):*

*rows(other.rows), cols(other.cols), vals(new double[rows\*cols])*

*{*

*for (int i=0; i<rows\*cols; ++i)*

*vals[i]=other.vals[i];*

*}*

ג [10 נקודות]. יש להוסיף קוד למחלקה PositivcMatrix כך שבשורה 37 יודפס הערך הנכון - 5.

*להוסיף אופרטור השמה:*

*PositiveMatrix& operator=(const PositiveMatrix& other)*

*{*

*if (this != &other)*

*{*

*delete[] vals;*

*rows = other.rows;*

*cols = other.cols;*

*vals = new double[rows\*cols];*

*for (int i=0; i<rows\*cols; ++i)*

*vals[i]=other.vals[i];*

*}*

*return \*this;*

*}*

ד [10 נקודות]. יש להוסיף ולשנות קוד כך שכל ניסיון לשנות ערך במטריצה למספר שלילי (כמו בשורה 39) יגרום לזריקת חריגה.

*להוסיף מחלקת פרוקסי ולהחזיר אותה מתוך אופרטור הסוגריים. למשל, אפשר להוסיף את הקוד הבא בשורה 5:*

*class negative\_value { };*

*class Proxy {*

*double& \_d;*

*public:*

*Proxy(double& d): \_d(d) { }*

*Proxy& operator= (double d) {*

*if (d<0)*

*throw negative\_value();*

*\_d = d;*

*return \*this;*

*}*

*operator double() const { return \_d; }*

*};*

*ולשנות את שורה 20 ל:*

*20* ***Proxy*** *operator() (int x, int y) {*

*21       return Proxy(vals[x\*cols+y]);*

*22 }*

## שאלה 7 [10 נק']

נתון קוד חלקי של המחלקה Fraction. עליכם לכתוב קוד חדש, כך שכל עצם מהמחלקה:

Fraction<int>

יתנהג כמו Fraction הנתון, אבל יהיה ניתן גם לייצר עצמים מהטיפוס:

Fraction<short>

Fraction<long>

בלי לכתוב קוד נוסף. אין צורך לממש אופרטורים או פונקציות אחרות שלא מופיעות בקוד להלן. עליכם לכתוב קוד מלא גם אם חלקו העתקה מהקוד להלן.

class Fraction {

private:

int nom, den;

public:

Fraction(int nn, int dn=1): nom(nn), den(dn) { }

int Nom() const {

return nom;

}

int Den() const {

return den;

}

Fraction operator+ (const Fraction& other) {

int nn = nom \* other.den + den \* other.nom;

int dd = den \* other.den;

return Fraction(nn, dd);

}

};

***פתרון****:*

*template <typename T>*

*class Fraction*

*{*

*private:*

*T nom, den;*

*public:*

*Fraction(T nn, T dn=1): nom(nn), den(dn) { }*

*T Nom() const {*

*return nom;*

*}*

*T Den() const {*

*return den;*

*}*

*Fraction operator+ (const Fraction& other) {*

*T nn = nom \* other.den +*

*den \* other.nom;*

*T dd = den \* other.den;*

*return Fraction(nn, dd);*

*}*

*};*