

פקולטה: מדעי הטבע

מחלקה: מדעי המחשב

שם הקורס: מבנה זיכרון ושפת ++C

קוד הקורס: 2-7027810 כל הקבוצות - קבוצות 1,3,4,5

מועד \_\_**ב**\_\_\_ סמסטר: \_\_\_ב\_\_\_ שנה: ה'תשעט

משך הבחינה: 3 שעות

שם המרצים: אראל סגל-הלוי, גיל בן-ארצי

יש לענות על כל השאלות במחברת הבחינה.

אין צורך להעתיק את השאלון למחברת - השאלון יתפרסם בגיטהאב לאחר הבחינה.

יש לענות תשובות מלאות אך ממוקדות. לא יינתנו נקודות על תשובות עם טקסט מיותר שאינו קשור לנושא.

אסור להשתמש בכל חומר עזר.

בהצלחה!!!

# **שאלה 1 [10 נק']**

01 #include <iostream>  
02 using **namespace** std;  
03  
04 class Base {  
05 public:  
06 Base(){}  
07 **void** Print( ){  
08 cout <<**" I am base "**<<endl;  
09 }  
10 Base& **operator**=(**const** Base & b){

11 return this;

12 }  
13 };  
14  
15 class Derived : **public** Base {  
16 public:  
17 Derived():Base(){  
18 }  
19  
20 **void** Print( ){  
21 cout <<**" I am derived "**<<endl;  
22 }  
23 Derived& **operator**=(**const** Derived &){  
24 return this;

25 }  
26 };

# **1.1 סעיפים א-ב [5 נק']**.

# עבור סעיפים א-ב, התייחסו לתוכנית הראשית הבאה:

int main() {  
 Derived d;  
 Base b;  
 b=d;  
 b.Print();  
 **return** 0;  
}

# א [3 נק']. בקוד יש שתי שגיאות קומפילציה. מה הן, ואיך אפשר לתקנן? אין למחוק קוד ואין לשנות את main.

# פתרון: בשורה 11 ובשורה 24 יש שגיאה – הפונקציה אמורה להחזיר רפרנס ומחזירה פוינטר.

# צריך להחזיר **\*this** במקום this.

# הערה: בשורה 23 אין שגיאה – לא חייבים לציין את שם הפרמטר לפונקציה אם לא משתמשים בו בגוף הפונקציה.

# ב [2 נק']. מה יהיה הפלט לאחר התיקון?

# פתרון: I am base

# **1.2 סעיפים ג-ד [5 נק']**.

# עבור סעיפים ג-ד, הניחו שהשגיאות מהסעיפים הקודמים תוקנו, והתייחסו לתוכנית הראשית הבאה:

int main() {  
 Derived d;  
 Base b;

b=d;  
 b.Print();  
 d=b;  
 d.Print();  
 **return** 0;  
}

# ג [3 נק']. בקוד יש שגיאת קומפילציה אחת. מה היא, ואיך אפשר לתקנה? אין לשנות את main.

# פתרון: בשורה החמישית של main, מנסים להמיר Base ל-Derived אבל האופרטור= מצפה לקבל Derived.

# תיקון: לשנות בשורה 23 את סוג הפרמטר ל Base&.

# ד [2 נק'] מה יהיה הפלט לאחר התיקון?

# I am base

# I am derived

# 

# **שאלה 2 [9 נק']**

01 #include <iostream>

02 #include <string>

03 using namespace std;

04

05 class Player {

06     public:

07         void play() {

08             cout << "Error: Player.play is not defined" << endl;

09         }

10 };

11

12 class SmartPlayer: public Player {

13     public:

14         void play() {

15             cout << "Smart Move" << endl;

16         }

17 };

18

19 class SimplePlayer: public Player {

20     public:

21         void play() {

22             cout << "Simple Move" << endl;

23         }

24 };

25

26 void game(Player\* player1, Player\* player2, int turns) {

27     for (int i=0; i<turns; ++i) {

28         player1->play();

29         player2->play();

30     }

31 }

32

33 int main() { // a demo program

34     game(new SmartPlayer(), new SimplePlayer(), 5);

35 }

כשמריצים את התוכנית, רואים 10 פעמים את המחרוזת:

Error: Player.play is not defined

א [3 נק']. מה סוג השגיאה (קומפילציה / קישור / זמן-ריצה / לוגית)? מה בדיוק גורם לשגיאה?

**פתרון**: שגיאה לוגית. הפונקציה game מקבלת פוינטר מסוג Player. כיוון שהפונקציה play לא וירטואלית, הקריאה בתוך הפונקציה game היא לplay של Player ולא של המחלקות היורשות.

ב [3 נק']. איך אפשר לתקן את הבאג ע"י שינוי/הוספה של שורה אחת בלבד? ציינו את מספר השורה שיש לשנות/להוסיף והסבירו מדוע השינוי פותר את הבעיה. *אין לשנות את התוכנית הראשית.*

**פתרון**: להוסיף virtual בשורה 7.

ג [3 נק']. מה יהיה פלט התוכנית לאחר התיקון?

**פתרון**:

Smart Move

Simple Move

Smart Move

Simple Move

Smart Move

Simple Move

Smart Move

Simple Move

Smart Move

Simple Move

# **שאלה 3 [8 נק']**

1. #include **<iostream>**
2. **using namespace** std;
3. **void** change(**int**\*& ip, **int** j) {
4. ip = (int \*)(++j);
5. }
6. **int** main() {
7. **int**\* ip = 0,j=0;
8. change(ip,j);
9. cout << **"ip = "** << ip << **" j="**<< (j++) <<endl;
10. cout << **"ip = "** << (++ip) << **" j="**<< j <<endl;
11. **return** 0;
12. }

א [4 נק']. האם יש שגיאת קומפילציה או אזהרה בתוכנית? אם כן ציינו היכן ותקנו אותה

פתרון: אין שגיאה – יש רק אזהרה על המרה משתמעת של int ל-int\*.

ב [4 נק']. מה פלט התוכנית אחרי התיקון (אם צריך)?

פתרון:

ip = 0x1 j=0

ip = 0x5 j=1

בשורה השניה זה 5 ולא 2, כי כשמקדמים פוינטר הוא מתקדם לפי מספר הבתים של מה שהוא מצביע עליו (במקרה זה מספר שלם – 4 בתים)

# **שאלה 4 [9 נק']**

נתונה התוכנית:

01 #include <iostream>

02 #include <algorithm>

03 #include <vector>

04 using namespace std;

05

06 struct Person {

07     string name;

08     int age;

09 };

10

11 int main() {

12     vector<Person> unsorted { {"bbb",11}, {"aaa",33}, {"ddd",22}, {"ccc",44} };

13     vector<Person> sorted\_by\_age = unsorted;

14    sort(sorted\_by\_age.begin(), sorted\_by\_age.end());

15 for (Person p: sorted\_by\_age)

16 cout << p.age << " ";

17 }

כשמקמפלים את התוכנית מתקבל פלט ארוך מאד, המתחיל ב:

/usr/bin/../lib/gcc/x86\_64-linux-gnu/7.3.0/../../../../include/c++/7.3.0/bits/predefined\_ops.h:43:23: **error**: invalid

operands to binary expression ('Person' and 'Person')

{ return \*\_\_it1 < \*\_\_it2; }

~~~~~~ ^ ~~~~~~

ומסתיים ב:

question.cpp:14:8: **note**: in instantiation of function template specialization

'std::sort<\_\_gnu\_cxx::\_\_normal\_iterator<Person \*, std::vector<Person, std::allocator<Person> > > >' requested here

sort(sorted\_by\_age.begin(), sorted\_by\_age.end());

^

א [3 נק']. מה סוג השגיאה? מדוע הודעת השגיאה (**error**) מפנה לקובץ שלא אנחנו כתבנו? ומדוע אחרי הודעת-השגיאה יש **note** שמפנה לקוד שאנחנו כתבנו?

**פתרון**: שגיאת קומפילציה. נגרמת כי אנחנו קוראים לאלגוריתם "sort" שצריך לסדר את התחום הנתון. לשם כך האלגוריתם צריך להשוות בין פרטים בתוך התחום. ההשוואה נעשית ע"י אופרטור "קטן מ" (<) אבל לא הגדרנו אופרטור כזה למחלקה Person. הקומפיילר מגלה את השגיאה לראשונה כשהוא מנסה ליצור מקרה פרטי (instantiation) של תבנית-הפונקציה sort, והקוד נמצא בספריה התקנית, ולכן ה-error הוא בקובץ של הספריה התקנית. כדי לעזור לנו לתקן את השגיאה, הקומפיילר כותב note שאומר לנו מאיפה התחיל ה-instantiation.

ב [3 נק']. הוסיפו קטע קוד *לפני התוכנית הראשית*, כך שהתוכנית תתקמפל ותרוץ והוקטור sorted\_by\_age אכן יכיל את רשימת האנשים מסודרים לפי גיל (מהצעיר למבוגר). *בסעיף זה אין לשנות את התוכנית הראשית.*

**פתרון**: יש להוסיף אופרטור "קטן מ-", למשל כך:

bool operator<(Person a,Person b) {

return a.age < b.age;

}

ג [3 נק']. הוסיפו קוד *בשורה 14 בלבד*, כך שהתוכנית תתקמפל ותרוץ והוקטור sorted\_by\_age אכן יכיל את רשימת האנשים מסודרים לפי גיל (מהצעיר למבוגר). *בסעיף זה אין לשנות שורות אחרות.*

**פתרון**: יש לשלוח ל-sort פרמטר שלישי שהוא פונקטור בינארי המייצג את הסדר הרצוי, למשל כך:

sort(sorted\_by\_age.begin(), sorted\_by\_age.end(), [](Person a, Person b){return a.age<b.age;});

# **שאלה 5 [22 נק']**

סטודנט כתב מחלקה בשם PhysicalNumber המייצגת מספר פיסיקלי עם יחידות, היכול לתמוך בחיבור וחיסור **וגם כפל וחילוק**. כיתבו בדיקות-יחידה מקיפות עבור המחלקה. יש לבדוק את הדרישות הבאות:

* תמיכה בארבעה אופרטורים חשבוניים + - \* / , וכן אופרטור הפלט.
* תמיכה בשני סוגים של יחידות אורך – מטר וקילומטר, ושני סוגים של יחידות זמן – שעה ודקה.
* כפל וחילוק אפשריים **תמיד** – בין כל סוגי היחידות. היחידות של התוצאה נקבעות לפי היחידות של המספרים המוכפלים/מחולקים. למשל, אם מחלקים 100 קילומטר ב-2 שעות, התוצאה היא "50 קילומטר / שעה". אם מכפילים "50 קילומטר / שעה" ב-30 דקות, התוצאה היא "25 קילומטר", וכו'.
* חיבור וחיסור – אפשרי רק בין מספרים עם יחידות תואמות, כגון: אורך+אורך, זמן+זמן, אורך/זמן + אורך/זמן, וכו'.
* חיבור וחיסור של מספרים עם יחידות לא-תואמות, למשל אורך + אורך/זמן, יגרום לחריגה.

א [11 נק']. **תיכנון הבדיקות**: כיתבו (בעברית) רשימה מפורטת של מקרים שאתם מתכוונים לבדוק.

פתרון: יש לבדוק את האופרטורים + - עם לפחות 12 סוגים שונים של משתנים, למשל מהסוגים הבאים:

* מטר, קילומטר, שעה, שניה
* מטר/שעה, מטר/שניה, קילומטר/שעה, קילומטר/שניה
* מטר\*שעה, מטר\*שניה, קילומטר\*שעה, קילומטר\*שניה
* מטר\*קילומטר/שעה, קילומטר\*שעה/שניה, שעה/קילומטר\*מטר, שניה\*מטר\*קילומטר
* צירופים נוספים מסוג זה..

ב [11 נק']. **מימוש הבדיקות**: מצורף קובץ עם בדיקות לדוגמה. עליכם להרחיב אותו ולהוסיף לו את הבדיקות בהתאם לתיכנון שכתבתם בסעיף א.

#include <iostream>

#include "badkan.hpp"

#include "PhysicalNumber.hpp"

using namespace std;

using ariel::PhysicalNumber, ariel::Unit;

int main() {

PhysicalNumber a(100, Unit::KM);

PhysicalNumber b(2, Unit::HOUR);

PhysicalNumber c(100, Unit::M);

PhysicalNumber d(30, Unit::MIN);

testcase

.setname("Multiplication and division")

.CHECK\_OUTPUT(a/b, "50[km/hr]") // 50 kilometers per hour

.CHECK\_OUTPUT(a\*b, "200[km\*hr]")

.CHECK\_OUTPUT(a\*b\*b, "400[km\*hr\*hr]")

.CHECK\_OUTPUT(a\*c, "10000[km\*m]")

.setname("Adding compatible units")

.CHECK\_OUTPUT(a/b + a/d, "250[km/hr]") // 50 km/hour + 200 km/hour

.CHECK\_OUTPUT(a/b - c/d, "49.8[km/hr]") // 50 km/hour - 200 m/hour

.setname("Adding incompatible units")

.CHECK\_THROWS(a/b + a) // a/b is [km/hour], a is [km]

.CHECK\_THROWS(a/b - a\*b) // a/b is [km/hour], a\*b is [km\*hour]

// ADD MORE TESTS HERE

.setname("...")

.print(cout);

}

# **שאלה 6 [24 נק']**

# נתון קוד של משחק בול-פגיעה (כמו במטלה 4):

# uint play(Chooser& chooser, Guesser& guesser, uint length, uint maxTurns) {

# const uint TECHNICAL\_VICTORY\_TO\_GUESSER = 0;

# const uint TECHNICAL\_VICTORY\_TO\_CHOOSER = maxTurns+1;

# string choice = chooser.choose(length);

# if (choice.length()!=length) // Illegal choice

# return TECHNICAL\_VICTORY\_TO\_GUESSER;

# guesser.startNewGame(length); // tell the guesser that a new game starts

# uint indexOfTurn;

# for (indexOfTurn=0; indexOfTurn<maxTurns; ++indexOfTurn) {

# string guess = guesser.guess();

# if (guess.length()!=length) // Illegal guess

# return TECHNICAL\_VICTORY\_TO\_CHOOSER;

# if (guess==choice) {

# return indexOfTurn + 1;

# } else {

# auto reply = calculateBullAndPgia(choice, guess);

# guesser.learn(reply);

# }

# }

# return TECHNICAL\_VICTORY\_TO\_CHOOSER; // Guesser could not guess in time

# }

# **הערה**: ניתן להניח שהקבצים Chooser.hpp, Guesser.hpp, calculate.hpp ושאר הקבצים ממטלה 4, כתובים בצורה נכונה – אין צורך לכתוב אותם מחדש.

# א [12 נק']. כיתבו מחלקה בשם ShuffleChooser, הבוחרת תמיד מספר המורכב מהספרות 1 עד length, כל ספרה פעם אחת, בסדר אקראי (ניתן להניח ש length קטן או שווה 9). למשל, אם length=3 ייבחר באקראי אחד המספרים 123,132,213,231,312,321.

# פתרון לדוגמה:

**ShuffleChooser.hpp:**

namespace bullpgia {

class ShuffleChooser: public Chooser {

public:

std::string choose(uint length);

};

}

**ShuffleChooser.cpp:**

namespace bullpgia {

std::string ShuffleChooser::choose(uint length) {

std::string s(length, ' '); // init a string with "length" spaces

std::iota(begin(s),end(s),'1');

std::shuffle(begin(s),end(s), std::random\_device{});

return s;

}

}

# ***מפתח ניקוד****:*

# *6 נקודות על ירושה וכותרות, 6 נקודות על המימושים.*

# *שימו לב - חשוב שתהיה ירושה מ-Chooser, אחרת הקריאה ל-play לא תעבוד.*

# ב [12 נק']. כיתבו מחלקה בשם PermutationGuesser, המנסה לנחש את המספר שבחר ShuffleChooser ע"י בדיקת כל הפרמוטציות האפשריות מ-1 עד length . למשל, אם length=3 המנחש יבדוק את כל 6 המספרים הנ"ל לפי הסדר עד שימצא את התשובה הנכונה (או עד שייגמר הזמן).

# לפניכם תוכנית ראשית לדוגמה:

# #include <iostream>

# using namespace std;

# #include "play.hpp"

# #include "ShuffleChooser.hpp"

# #include "PermutationGuesser.hpp"

# int main() {

# bullpgia::ShuffleChooser sc;

# bullpgia::PermutationGuesser pg;

# cout << bullpgia::play(sc, pg, 3, 100) << endl;

# // ShuffleChooser chooses a random permutation of 1,2,3

# // PermutationGuesser guesses all of them until the correct one is found.

# cout << bullpgia::play(sc, pg, 9, 100) << endl;

# // ShuffleChooser chooses a random permutation of 1,...,9.

# // PermutationGuesser guesses all of them until the correct one is found.

# return 0;

# }

# פתרון לדוגמה:

PermutationGuesser.hpp:

namespace bullpgia {

    class PermutationGuesser: public Guesser {

        protected:

            std::string next\_guess;

        public:

            void startNewGame(uint length);

            std::string guess();

            void learn(const Reply& reply);

    };

}

PermutationGuesser.cpp:

namespace bullpgia {

            void PermutationGuesser::startNewGame(uint length) {

next\_guess.resize(length);

std::iota(begin(next\_guess),end(next\_guess),'1');

            }

            std::string PermutationGuesser::guess() {

                std::next\_permutation(begin(next\_guess),end(next\_guess));

                return next\_guess;

            }

            void PermutationGuesser::learn(const Reply& reply) { }

}

# ***מפתח ניקוד****:*

# *6 נקודות על ירושה וכותרות, 6 נקודות על המימושים.*

# *שימו לב – חשוב לממש את כל שלוש השיטות הנזכרות בפונקציה play – שהן guess, learn, startNewGame .*

# 

# **שאלה 7 [22 נק']**

כיתבו מחלקה בשם **binop**, שהבנאי שלה מקבל שלושה קלטים:

* שני דמויי-מיכלים (iterables) באותו אורך;
* פונקטור בינארי,

ובונה דמוי-מיכל (iterable) המתקבל מהפעלת הפעולה הבינארית על זוגות איברים תואמים בשני הקלטים. אם הקלטים אינם באותו אורך – ההתנהגות לא מוגדרת. לפניכם תוכנית לדוגמה.

#include <iostream>

#include <vector>

#include <list>

using namespace std;

#include "range.hpp"

#include "binop.hpp"

using namespace itertools;

int main() {

    vector<int> v1{10,20,30,40,50};

    list<double> L1{1.1, 2.2, 3.3, 4.4, 5.5};

    for (auto i: binop(v1, L1, [](int x,double y){return x+y;} ))

        cout << i << " "; // Prints 11.1 22.2 33.3 44.4 55.5

         // since 11.1 = 10 + 1.1, 22.2 = 20 + 2.2, etc.

    cout << endl;

    for (auto i: binop(v1, range(1,6), [](int x,int y){return x\*y;} ))

        cout << i << " "; // Prints 10 40 90 160 250

         // since 10\*1 = 10, 20\*2 = 40, etc.

    cout << endl;

    return 0;

}

לפניכם התחלה של מימוש המחלקה בקובץ binop.hpp. השלימו את החסר.

namespace itertools {

    template<typename T1, typename T2, typename OP>

class binop {

...

    };

}

**שימו לב**: אתם צריכים לכתוב רק את הקובץ binop.hpp. אתם **לא צריכים** לכתוב את הקובץ range.hpp – אפשר להניח שהוא כתוב ועובד נכון כמו במטלה. אין להניח הנחות נוספות.

**פתרון אפשרי**: בעמוד הבא.

מפתח ניקוד: 11 נקודות על הגדרה נכונה של הכותרות (בנאי, איטרטור, \*, ++, ==, !=, begin, end):

11 נקודות על מימושים נכונים.

namespace itertools {

    template<typename T1, typename T2, typename OP>     class binop {

public:

        const T1 first;

        const T2 second;

        const OP op;

        binop(const T1& first, const T2& second, OP op): first(first), second(second), op(op) {}

        struct iterator {

            decltype(first.begin()) firstIterator, firstEnd;

            decltype(second.begin()) secondIterator, secondEnd;

            const OP op;

            auto operator\*() {

                return op(\*firstIterator, \*secondIterator);

            }

            iterator& operator++() {

                ++firstIterator;

                ++secondIterator;

                return \*this;

            }

            iterator operator++(int) {

                iterator tmp=\*this;

                operator++();

                return tmp;

            }

            bool operator==(const iterator& other) {

                return firstIterator==other.firstIterator && secondIterator==other.secondIterator;

            }

            bool operator!=(const iterator& other) {

                return !(operator==(other));

            }

        };

        iterator begin() const {

            return iterator{first.begin(), first.end(), second.begin(), second.end(), op};

        }

        iterator end() const {

            return iterator{first.end(), first.end(), second.end(), second.end(), op};

        }

    };

}

# **שאלה 8 - מענקים**

א [3 נק']. האם השתמשנו בבדיקות שלכם באחת המטלות? אם כן אנא ציינו באיזו מטלה בדיוק.

ב [2-4 נק']. האם זכיתם באחד משלושת המקומות הראשונים בתחרות בול-פגיעה? אם כן ציינו באיזה מקום.