**מועד ב חורף 2016 פתרון**

**שאלה 1 - ADT**

1. פתרון

PSet SetDuplicate(PSet pSet)

{

int i;

PSet pNewSet = CreateSet(pSet);

for (i = 0; i < SetGetSize(pSet); i++)

{

SetAdd(pNewSet, SetGetElem(pSet, NULL));

}

return pNewSet;

}

1. פתרון

PSet SetUnion(PSet pA, PSet pB)

{

int i;

PSet pC = SetDuplicate(pA);

for (i = 0; i < SetGetSize(pB); i++)

{

SetAdd(pC, SetGetElem(pB, NULL));

}

return pC;

}

1. פתרון

PSet SetIntersect(PSet pA, PSet pB)

{

int i;

PSet pC = CreateSet(pA);

PSet pTmpSet = SetDuplicate(pA);

PElement pElem;

for (i = 0; i < SetGetSize(pB); i++)

{

pElem = SetGetElem(pB, NULL);

if (SetAdd(pTmpSet, pElem) == FAIL)

SetAdd(pC, pElem);

}

DeleteSet(pTmpSet); // free tmp set

return pC;

}

1. פתרון

Result IsAContainedInB(PSet pA, PSet pB)

{

PSet pC = SetIntersect(pA, pB);

Result res = (SetGetSize(pA) == SetGetSize(pC)) ? SUCCESS : FAIL;

DeleteSet(pC);

return res;

}

1. פתרון

PElement SetOfSetCopy(const PElement pElem)

{

return SetDuplicate((PSet)pElem);

}

void SetOfSetDel(PElement pElem)

{

DeleteSet((PSet)pElem);

}

PKey SetOfSetGetKey(PElement pElem)

{

return (PKey)pElem;

}

Result SetOfSetIsEq(const PKey pKey1, const PKey pKey2)

{

if (IsAContainedInB((PSet)pKey1, (PSet)pKey2) == SUCCESS && IsAContainedInB((PSet)pKey2, (PSet)pKey1) == SUCCESS)

return SUCCESS;

return FAIL;

}

1. פתרון

pSetOfSets = CreateSet(SetOfSetCopy, SetOfSetDel, SetOfSetGetKey, SetOfSetIsEq);

1. פתרון

typedef struct MapPair\_

{

PKey Pkey;

PElement pData;

}MapPair, \*PMapPair;

1. פתרון

PElement MapCopyFunc(PElement pElem)

{

PMapPair pCopy;

PMapPair pMapPair = (PMapPair)pElem;

pCopy = (PMapPair)malloc(sizeof(MapPair));

if (pCopy == NULL)

exit(-1);

pCopy->Pkey = DoubleCopy(pMapPair->Pkey);

pCopy->pData = DoubleCopy(pMapPair->pData);

return pCopy;

}

void MapDelFunc(PElement pElem)

{

PMapPair pMapPair = (PMapPair)pElem;

if (pElem == NULL)

return;

DoubleDel(pMapPair->pData);

DoubleDel(pMapPair->Pkey);

free(pMapPair);

}

PKey MapGetKeyFunc(PElement pElem)

{

return ((PMapPair)pElem)->Pkey;

}

1. פתרון

PMap CreateMap()

{

PMap pMap = (PMap)malloc(sizeof(Map));

if (pMap == NULL)

exit(-1);

pMap->pSet = CreateSet(**MapCopyFunc**, **MapDelFunc**, **MapGetKeyFunc**, **DoubleIsEq**);

return pMap;

}

Result MapAdd(PMap pMap, PKey pKey, PElement pAddElem)

{

PMapPair oldPair;

MapPair mapPair = { pKey, pAddElem };

if (pKey == NULL || pAddElem == NULL)

return FAIL;

return SetAdd(pMap->pSet, (PElement)&mapPair);

}

שאלה 2 פתרון

1. **פתרון:**

class Chord

{

public:

Chord();

Chord(const Chord& chord);

void operator+=(const char\* note);

Chord& operator=(const Chord& chord);

void Print() const;

std::vector<char\*> GetNotes();

~Chord();

private:

std::vector<char\*> m\_notes;

friend Chord operator+(Chord& chord, char\* note);

friend Chord operator+(char\* note, Chord& chord);

};

Chord operator+(Chord& chord, char\* note);

Chord operator+(char\* note, Chord& chord);

1. **פתרון:**

void Chord::operator+=(const char \* note)

{

if (std::strlen(note) == 0)

throw note;

if (note[0] < 'A' || note[0] > 'G')

throw note;

char\* newNote = new char[std::strlen(note) + 1];

std::strcpy(newNote, note);

m\_notes.push\_back(newNote);

}

1. **פתרון:**

class Instrument

{

public:

Instrument(const char\* lowestNote, const char\* highestNote);

virtual void Play(Chord chord);

~Instrument();

protected:

char \*m\_lowestNote, \*m\_highestNote;

};  
  
  
class Piano :

public Instrument

{

public:

Piano(char\* lowestNote = "A0", char\* highestNote = "B7");

virtual void Play(Chord chord);

~Piano();

};

class Guitar :

public Instrument

{

public:

Guitar(char\* lowestNote = "E2", char\* highestNote = "E4", int numStrings

= 6);

virtual void Play(Chord chord);

~Guitar();

private:

int m\_numStrings;

};

class Part

{

public:

Part(Instrument\* instrument, int tempo = 120, double meter = 1.0);

void operator+=(const Chord& chord);

void Play();

void Print();

~Part();

private:

std::vector<Chord> m\_chords;

Instrument\* m\_instrument;

int m\_tempo;

double m\_meter;

};

1. **פתרון:**

void PrintChord(const Chord& chord)

{

chord.Print();

}

void Part::Print()

{

std::for\_each(m\_chords.begin(), m\_chords.end(), PrintChord);

}

# שאלה 3 – C++ ( 20 נק')

1. קודם כל אין שחרור של הזכרון המוקצה. מעבר לכך at זורקת חריגה כשניגשים לאינדקס לא חוקי (3<) ולכן יש זליגת זכרון.

דגש: חייבים לעטוף את הלולאה בtry catch שיתפוס את הזריקה ובנוסף יש לשחרר את הזכרון שהוקצה.

cout << "There are " << Person\_DB.size() << " people in the data base." << endl;

try {

for (int i=0;i<10;i++) {

Person\_DB.at(i)->print();

}

}

catch(const out\_of\_range& e) {

cout << "Can't print Person\_DB: Out of Range - " << e.what() <<endl;

}

for (vector<Person\*>::iterator pit = Person\_DB.begin(); pit != Person\_DB.end(); ++pit)

delete \*pit;

2)

(1) יש להגדיר בתור ידידה את ה function object

(2) יש להגדיר function object שחופף את operator() וממש את יחס הסדר בין הסטודנטים

(3) יש לשפוך את תוכן מבנה הנתונים מסוג map לתוך vector, להפעיל את פונקציית sort עליו ואח"כ לעובר בלולאה על הוקטור הממויין ולהדפיס.

class Student {

public:

Student(const char\* Name, int Grade);

~Student();

void Print() const;

**(1)** friend class compare\_students ;

private:

char \*Name\_;

int Grade\_;

};

Student::Student(const char\* Name, int Grade) : Grade\_(Grade) {

Name\_ = new char[strlen(Name)+1];

strcpy(Name\_,Name);

}

Student::~Student(){

delete [] Name\_;

}

void Student::Print() const{

cout << Name\_ << endl;

cout << Grade\_ << endl << endl;

}

**(2)** class compare\_students {

public:

bool operator()(Student\* a, Student\* b) {

return a->Grade\_ > b->Grade\_ ? true:false;

};

};

int main() {

map<unsigned int, Student\*> Smap;

Smap.clear();

Smap[123456789] = new Student("John Doe", 85);

Smap[987654321] = new Student("Uncle Sam", 80);

Smap[666666666] = new Student("Freddy Kruger", 66);

Smap[246813579] = new Student("Master Splinter", 100);

**(3)** vector<Student\*> Student\_vec;

Student\_vec.clear();

for (map<unsigned int, Student\*>::iterator it = Smap.begin();

it != Smap.end(); ++it) {

Student\_vec.push\_back((it->second));

}

compare\_students cs;

sort(Student\_vec.begin(), Student\_vec.end(), cs);

for (vector<Student\*>::iterator it = Student\_vec.begin();

it != Student\_vec.end(); ++it) {

(\*it)->Print();

}

for (map<unsigned int, Student\*>::iterator it = Smap.begin();

it != Smap.end(); ++it) {

delete it->second;

}

}

**שאלה 4 פתרון**

**שאלת BASH**

סעיף א':

**grades\_in\_range**:

#!/bin/bash  
cut -f1,$2 -d, $1 | grades\_in\_range\_aux $3 $4

**grades\_in\_range\_aux**:

#! /bin/bash  
read line  
echo $line  
while read line; do  
line\_split=($line)  
if (( ${line\_split[1]}>=$1 && ${line\_split[1]}<=$2 )); then  
echo $line  
fi  
done

סעיף ב':

**grades\_of\_task:**

#!/bin/bash  
titles=(`head -n 1 $1`)  
ind=1  
while (( $ind < ${#titles[@]}+1 )); do  
title=`head -n 1 $1 | cut -f$ind -d,`  
if [[ $title == " $2" ]]; then  
grades\_in\_range $1 $ind $3 $4  
break  
fi  
(( ind=$ind+1 ))  
done

**שאלות הבנה:**

סעיף 2: שתי בעיות:

* אופרטור [] מקבל reference, לכן אי אפשר להעביר לו 1 כי זה קבוע. תיקון: אופרטור [] חייב לקבל value או const reference, כלומר int או const int&.
* ב. אופרטור [] מחזיר by value, לכן אין אפשרות לעשות השמה ל- c1[1]. תיקון: אופרטור [] חייב להחזיר by reference, כלומר double&.

סעיף 3: שני אופרטורי + מחזירים reference למשתנה לוקלי שלהם שנמחק עם סיום ביצוע האופרטור, מה שיגרום להתנהגות לא צפויה. תיקון: שני אופרטורי + חייבים להחזיר by value, כלומר להחזיר Complex.

סעף 4: אין בעיה, השורה תתבצע כמצופה. יקרא constructor שימיר 5 ל- Complex ולאחר מכן יקראו שני אופרטורי +: אופרטור member שמחבר בין שני מרוכבים ואופרטור חיצוני שמחבר בין double ו- Complex.