מבחן מועד X - פתרון

במבחן 11 שאלות. משקל כל שאלה הוא 10 נקודות.
הציון יהיה על 10 השאלות עם הציון הגבוה ביותר.

- עבור השאלות העוסקות בקומפילציה של הקוד, יש להניח שאין שגיאות סינטקטיות, אם קיימת שגיאת קומפילציה היא מהותית (לא חוסר ב;)
- במידה ומצאתם שגיאת קומפילציה והצעתם תיקון לקוד יש לענות על יתר השאלות ביחס לקוד המתוקן.

השאלות יתייחסו לקטע הקוד הבא:

```
#include <iostream>
using namespace std;

class Sandwich
{
public:
	enum class eBreadType { WHITE, CHIFFON, FULL };
	enum class eStatus { NOT_EATEN, BEING_EATEN, THROWN_AWAY, FINISHED };
	static const char* breadTypes[];
	static const char* statusTypes[];
	Sandwich(const char* spread, eBreadType breadType = eBreadType::FULL);
	Sandwich(const Sandwich& other);
	Sandwich(Sandwich&& other);
	Sandwich(eBreadType breadType = eBreadType::FULL);
	const Sandwich& operator=(const Sandwich& other);
	const Sandwich& operator=(Sandwich&& other);
	~Sandwich();
	void setStatus(Sandwich::eStatus newStatus) const { status = newStatus;
}
	friend ostream& operator<<(ostream& os, const Sandwich& other);
	operator const char*() const { return spread; }
private:
	char* spread;
	eBreadType breadType;
	mutable eStatus status;


const char* Sandwich::breadTypes[] = { "White", "Chiffon", "Full" };
const char* Sandwich::statusTypes[] = { "not eaten", "being eaten",
"thrown away", "finished" };

Sandwich::Sandwich(const char* spread, eBreadType breadType)
{
	this->spread = _strdup(spread);
	this->breadType = breadType;
	status = eStatus::NOT_EATEN;
	cout << "In Sandwich::Sandwich with " << *this << endl;
}
Sandwich::Sandwich(const Sandwich& other)
{
	this->spread = _strdup(other.spread);
	this->breadType = other.breadType;
	status = eStatus::NOT_EATEN;
	cout << "In Sandwich::Sandwich (copy) for " << *this << endl;
}
Sandwich::Sandwich(Sandwich&& other)
{
```

```
	cout << "In Sandwich::Sandwich (move) for " << *this << endl;
	spread = other.spread;
	other.spread = nullptr;
	breadType = other.breadType;
	status = eStatus::NOT_EATEN;
}

Sandwich::Sandwich(eBreadType breadTyp) : spread(nullptr)
{
	cout << "In Sandwich::Sandwich (defualt)" << endl;
}
const Sandwich& Sandwich::operator=(const Sandwich& other)
{
	cout << "In Sandwich::operator= for " << *this << endl;
	if (this != &other)
	{
		delete[] spread;
		spread = _strdup(other.spread);
		breadType = other.breadType;
		status = other.status;
	}
	return *this;
}
const Sandwich& Sandwich::operator=(Sandwich&& other)
{
	cout << "In Sandwich::operator=&& for " << *this << endl;
	if (this != &other)
	{
		delete[] spread;
		spread = other.spread;
		other.spread = nullptr;
		breadType = other.breadType;
		status = other.status;
	}
	return *this;
}
Sandwich::~Sandwich()
{
	cout << "In Sandwich::~Sandwich for " << *this << endl;
	delete[]spread;
}

ostream & operator<<(ostream& os, const Sandwich& other)
{
	if (other.spread)
		os << other.spread;
	else
		os << "NO SPREAD";
	os << " in " << Sandwich::breadTypes[(int)other.breadType]
		<< " bread, status: " <<
Sandwich::statusTypes[(int)other.status];
	return os;
}

class SchoolBag
{
public:
	SchoolBag(const char* name, const Sandwich& sandwich = "cottege");
	SchoolBag(const char* name, Sandwich&& sandwich = "yellow cheese");
	SchoolBag(Sandwich&& sandwich, const char* name = "lolo");
	SchoolBag(const SchoolBag& other);
	const SchoolBag& operator=(const SchoolBag& other);
```

```
	~SchoolBag();
private:
	char* name;
	Sandwich sandwich;
};

SchoolBag::SchoolBag(const char* name, const Sandwich& sandwich) :
sandwich(sandwich)
{
	cout << "In SchoolBag::SchoolBag (ver 1) for " << name << "\n";
	this->name = _strdup(name);
}
SchoolBag::SchoolBag(const char* name, Sandwich&& sandwich) :
sandwich(sandwich)
{
	cout << "In SchoolBag::SchoolBag (ver 2) for " << name << "\n";
	this->name = _strdup(name);
}
SchoolBag::SchoolBag(Sandwich&& sandwich, const char* name) :
sandwich(std::move(sandwich))
{
	cout << "In SchoolBag::SchoolBag (ver 3) for " << name << "\n";
	this->name = _strdup(name);
}
SchoolBag::SchoolBag(const SchoolBag& other)
{
	Sandwich new_sandwich("Tahini", Sandwich::eBreadType::CHIFFON);
	sandwich = new_sandwich;
	cout << "In SchoolBag::SchoolBag (copy) for " << other.name << "\n";
	this->name = _strdup(other.name);
}

const SchoolBag& SchoolBag::operator=(const SchoolBag& other)
{
	cout << "In SchoolBag::operator= for " << name << "\n";
	if (this != &other)
	{
		delete[] name;
		name = _strdup(other.name);
		sandwich = other.sandwich;
	}
	return *this;
}
SchoolBag::~SchoolBag()
{
	cout << "In SchoolBag::~SchoolBag for " << name << "\n";
	delete[]name;
}


Sandwich prepareSandwich(const char* spread, Sandwich::eBreadType breadType)
{
	return Sandwich(spread, breadType);
}

void main1()
{
	int numOfSandwiches = 4;
	Sandwich** sandwiches = new Sandwich*[numOfSandwiches];

	Sandwich s1 = { "chocolate", Sandwich::eBreadType::WHITE };
	sandwiches[0] = &s1;
```

```
        sandwiches[1] = new Sandwich(*sandwiches[0]);
        sandwiches[1]->setStatus(Sandwich::eStatus::BEING_EATEN);
        const Sandwich s3("cheese");
        sandwiches[2] = new Sandwich(s3);
        sandwiches[3] = sandwiches[1];

        sandwiches[1]->setStatus(Sandwich::eStatus::BEING_EATEN);

        cout << "1) ----------------------------------\n";
        for (int i = 0; i < numOfSandwiches; i++)
        {
                cout << *(sandwiches[i]) << endl;
        }

        cout << "2) ----------------------------------\n";

        for (int i = 0; i < numOfSandwiches; i++)
        {
                delete sandwiches[i];
        }
        delete[]sandwiches;
}

void main2()
{
        Sandwich s("chocolate");
        SchoolBag* sb1 = new SchoolBag("koko",
                prepareSandwich("jam", Sandwich::eBreadType::CHIFFON));
        cout << "1) ----------------------------------\n";
        s = prepareSandwich("humus", Sandwich::eBreadType::WHITE);
        cout << "2) ----------------------------------\n";
        SchoolBag sb2("momo", Sandwich("cheese", Sandwich::eBreadType::FULL));
        cout << "3) ----------------------------------\n";
        SchoolBag sb3(Sandwich("nutella", Sandwich::eBreadType::WHITE));
        cout << "4) ----------------------------------\n";
        SchoolBag sb4(sb3);
        SchoolBag& sb5 = sb4;
        cout << "5) ----------------------------------\n";
        delete sb1;
}
```

- **שאלות 1-4 מתייחסות לmain1.**

שאלה 1:

בדוק האם main1 תתקמפל בצורה תקינה.
במידה וכן ציין זאת והמשך לשאלות הבאות.
במידה ולא הסבר מדוע לא והצע קוד אחר אשר ימנע את הבעיה, יש להיעזר בקווים המודפסים כדי לציין היכן יש להסיר או להוסיף קוד.

כן יש נסיון לשחרר זכרון עבור משתנים שכבר שוחררו מכיוון שיותר ממצביע אחד מצביע לאותו האובייקט.
יש להחליף את לולאת הfor שלאחר קו 2 בקוד הבא:

```
        delete sandwiches[1];
        delete sandwiches[2];
```

שאלות 2-4 :

כפתרון לשאלה 2 התייחסו לקוד עד הדפסת הקו שמתחיל ב- 1

```
In Sandwich::Sandwich with chocolate in White bread, status: not eaten
In Sandwich::Sandwich (copy) for chocolate in White bread, status: not eaten
In Sandwich::Sandwich with cheese in Full bread, status: not eaten
In Sandwich::Sandwich (copy) for cheese in Full bread, status: not eaten
```

כפתרון לשאלה 3 התייחסו לקוד בין הדפסות הקווים 1 ל- 2

chocolate in White bread, status: not eaten
chocolate in White bread, status: being eaten
cheese in Full bread, status: not eaten
chocolate in White bread, status: being eaten

כפתרון לשאלה 4 התייחסו לקוד לאחר הדפסת הקו שמתחיל ב- 2
In Sandwich::~Sandwich for chocolate in White bread, status: being eaten
In Sandwich::~Sandwich for cheese in Full bread, status: not eaten
In Sandwich::~Sandwich for cheese in Full bread, status: not eaten
In Sandwich::~Sandwich for chocolate in White bread, status: not eaten

- **שאלות 5-11 מתייחסות לmain2.**

שאלה 5:

בדוק האם 2main תתקמפל בצורה תקינה.
במידה וכן ציין זאת והמשך לשאלות הבאות.
במידה ולא הסבר מדוע לא והצע קוד אחר אשר ימנע את הבעיה, יש להיעזר בקווים המודפסים כדי לציין היכן יש להסיר או להוסיף קוד.

אין שגיאת קומפילציה

שאלות 6-11:

כפתרון לשאלה 6 התייחסו לקוד עד הדפסת הקו שמתחיל ב- 1
In Sandwich::Sandwich with chocolate in Full bread, status: not eaten
In Sandwich::Sandwich with jam in Chiffon bread, status: not eaten
In Sandwich::Sandwich (copy) for jam in Chiffon bread, status: not eaten
In SchoolBag::SchoolBag (ver 2) for koko
In Sandwich::~Sandwich for jam in Chiffon bread, status: not eaten
כפתרון לשאלה 7 התייחסו לקוד בין הדפסות הקווים 1 ל- 2
In Sandwich::Sandwich with humus in White bread, status: not eaten
In Sandwich::operator=&& for chocolate in Full bread, status: not eaten
In Sandwich::~Sandwich for NO SPREAD in White bread, status: not eaten
כפתרון לשאלה 8 התייחסו לקוד בין הדפסות הקווים 2 ל- 3
In Sandwich::Sandwich with cheese in Full bread, status: not eaten
In Sandwich::Sandwich (copy) for cheese in Full bread, status: not eaten
In SchoolBag::SchoolBag (ver 2) for momo
In Sandwich::~Sandwich for cheese in Full bread, status: not eaten
כפתרון לשאלה 9 התייחסו לקוד בין הדפסות הקווים 3 ל- 4
In Sandwich::Sandwich with nutella in White bread, status: not eaten
In Sandwich::Sandwich (move) for NO SPREAD in White bread, status: not eaten
In SchoolBag::SchoolBag (ver 3) for lolo
In Sandwich::~Sandwich for NO SPREAD in White bread, status: not eaten
כפתרון לשאלה 10 התייחסו לקוד בין הדפסות הקווים 4 ל- 5
In Sandwich::Sandwich with nutella in White bread, status: not eaten
In Sandwich::Sandwich (move) for NO SPREAD in White bread, status: not eaten
In SchoolBag::SchoolBag (ver 3) for lolo
In Sandwich::~Sandwich for NO SPREAD in White bread, status: not eaten
כפתרון לשאלה 11 התייחסו לקוד לאחר הדפסת הקו שמתחיל ב- 5
In SchoolBag::~SchoolBag for koko
In Sandwich::~Sandwich for jam in Chiffon bread, status: not eaten
In SchoolBag::~SchoolBag for lolo
In Sandwich::~Sandwich for Tahini in Chiffon bread, status: not eaten
In SchoolBag::~SchoolBag for lolo
In Sandwich::~Sandwich for nutella in White bread, status: not eaten
In SchoolBag::~SchoolBag for momo
In Sandwich::~Sandwich for cheese in Full bread, status: not eaten
In Sandwich::~Sandwich for humus in White bread, status: not eaten