מבחן מועד Y - שאלון

במבחן 12 שאלות. משקל כל שאלה הוא 10 נקודות.

הציון יהיה על 10 השאלות עם הציון הגבוה ביותר.

עבור כל שאלה יש להתייחס לדברים הבאים:

- ✓ במידה והיא מתקמפלת יש לכתוב מהו הפלט, כולל מה שיודפס לאחר הסוגריים המסולסלים שסוגרים את התוכנית
- ✓ במידה והשאלה אינה מתקמפלת, יש לכתוב מדוע
- ✓ במידה וישנה תעופה מהקוד, יש לכתוב מהו הפלט עד התעופה ולהסביר מדוע הקוד עף
- ✓ במידה ויש קוד שלא כל הזיכרון שוחרר, יש לציין זאת

<u>שאלות 1 עד 6 יתייחסו לקטע הקוד הבא:</u>

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

template<class T>
class Singleton
{
protected:
	Singleton() {} // what would happen if instead {} we wrote ;

	Singleton(const Singleton& other);
	const Singleton& operator=(const Singleton& other);

	static T theSingleton;

public:
	static T& getInstance() { return theSingleton; }
};

template<class T>
T Singleton<T>::theSingleton;

class Student
{
	char name[10];
	int id;

public:
	Student(const char* name, int id)
	{
		strcpy(this->name, name);
		this->id = id;
	}
};
```

```
class College : public Singleton<College>
{
	Student* allStudents[10];
	int numOfStudents;

	College(const College& other) = delete;
	College() = default;

	friend class Singleton<College>;

public:
	bool addStudent(Student& theStudent)
	{
		if (numOfStudents == 10)
			return false;

		allStudents[numOfStudents++] = &theStudent;
		return true;
	}

	operator int() const { return numOfStudents; }
};

int main()
{
	College& c1 = Singleton<College>::getInstance();
	College& c2 = Singleton<College>::getInstance();

	Student s1("gogo", 111);
	Student s2("momo", 222);
	c1.addStudent(s1);
	c1.addStudent(s2);

	Student s3("koko", 333);
	c2.addStudent(s3);

	cout << c1 << endl;
	cout << c2 << endl;
}
```

שאלה 1:

מה יקרה בעקבות הרצת ה- main?

שאלה 2:

נחליף כעת ב- main הנ"ל את הגדרת המשתנים c1 ו- c2 עם ההגדרה הבאה:

```
College c1 = Singleton<College>::getInstance();
College c2 = Singleton<College>::getInstance();
```

מה יקרה כעת בעקבות הרצת ה- main?

שאלה 3:

נחליף כעת ב- main הנ"ל את הגדרת המשתנים c1 ו- c2 עם ההגדרה הבאה:

```
College c1;
College c2;
```

מה יקרה כעת בעקבות הרצת ה- main?

שאלה 4:

במחלקה Singleton הקונסטרקטור מוגדר כך:

```
Singleton() {}
```

מאחר והמחלקה Singleton הינה מחלקת template, מישהו הציע להחליף את השורה כך שתראה כך, ללא מימוש בהמשך:

```
Singleton();
```

מה יקרה כעת בעקבות הרצת ה- main המקורי?

שאלה 5:

המחלקה College נותנת חברות למחלקה Singleton. האם הדבר הכרחי? נמק.

שאלות 6 עד 12 יתייחסו לקטע הקוד הבא:

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

class Ink
{
	bool isBlack;
	char* manufacturer;

	const Ink& operator=(const Ink& other);
public:
	Ink(const char* manufacturer, bool isBlack)
	{
		this->manufacturer = strdup(manufacturer);
		this->isBlack = isBlack;
		cout << "Creating ink: " << *this << endl;
	}

	Ink(const Ink& other)
	{
		manufacturer = strdup(other.manufacturer);
		isBlack = other.isBlack;
		cout << "Copying ink: " << *this << endl;
	}

	Ink(Ink&& other) : manufacturer(nullptr)
	{
		swap(manufacturer, other.manufacturer);
		isBlack = other.isBlack;
		cout << "Moving ink: " << *this << endl;
	}

	~Ink()
	{
		cout << "Deleting ink: " << *this << endl;
		delete[]manufacturer;
	}

	friend ostream& operator<<(ostream& os, const Ink& i)
	{
		if (i.manufacturer)
			os << i.manufacturer;
		else
			os << "Manufacturer already deleted";

		os << ", " << (i.isBlack ? "black" : "color") << " ink";
		return os;
	}

	const char* getManufacturer() const { return manufacturer; }
};
```

```
class Printer
{
	Ink theInk;
	char* manufacturer;
	bool isScanner;
	int serialNumber;

	static int serialNumberGenerator;

	const Printer& operator=(const Printer& other);
public:
	Printer(const char* manufacturer, const Ink& theInk, bool isScanner = true)
	                                                          : theInk(theInk)
	{
		this->manufacturer = strdup(manufacturer);
		this->isScanner = isScanner;
		serialNumber = ++serialNumberGenerator;
		cout << "Creating Printer: " << *this << endl;
	}

	Printer(const char* manufacturer, Ink&& theInk, bool isScanner = true)
	                                          : theInk(move(theInk))
	{
		this->manufacturer = strdup(manufacturer);
		this->isScanner = isScanner;
		serialNumber = ++serialNumberGenerator;
		cout << "Creating Printer with move ink: " << *this << endl;
	}

	Printer(const Printer& other) : theInk(other.theInk)
	{
		manufacturer = strdup(other.manufacturer);
		isScanner = other.isScanner;
		serialNumber = ++serialNumberGenerator;
		cout << "Copying Printer: " << *this << endl;
	}

	Printer(Printer&& other) : theInk(move(other.theInk))
	{
		swap(manufacturer, other.manufacturer);
		isScanner = other.isScanner;
		serialNumber = serialNumberGenerator;
		cout << "Moving Printer: " << *this << endl;
	}

	~Printer()
	{
		cout << "Deleting Printer: " << *this << endl;
		delete[]manufacturer;
	}

	friend ostream& operator<<(ostream& os, const Printer& p)
	{
		os <<  p.manufacturer
			<< ", S/N: " << p.serialNumber
			<< ", " << (p.isScanner ? "also" : "not") << " a scanner. "
			<< "Ink: " << p.theInk
			<< (strcmp(p.theInk.getManufacturer(), p.manufacturer) == 0 ?
                                                  " (original ink)" : "");
		return os;
	}
};
```

```
int Printer::serialNumberGenerator = 1000;

Printer createPrinterWithOrigInk(const char* manufacturer, bool isBlackInk,
                                 bool isScanner)
{
       return Printer(manufacturer, Ink(manufacturer, isBlackInk), isScanner);
}

void mainB1()
{
       Ink i1("hp", true);
       Printer p1("hp", i1);
       cout << "1 ----------------------\n";

       Printer p2("epson", Ink("hp", true), false);
       cout << "2 ----------------------\n";

       Printer p3(p1);
       cout << "3 ----------------------\n";

       cout << createPrinterWithOrigInk("epson", false, true) << endl;
       cout << "4 ----------------------\n";
}

void mainB2()
{
       Ink i1("hp", true);
       Ink i2("epson", false);

       i1 = i2;
}

void mainB3()
{
       Ink i1("hp", true);
       Printer p1("hp", i1);
       Printer p2("epson", Ink("hp", true), false);

       p1 = p2;
}
```

שאלות 6 – 10:

מה יקרה בעקבות הרצת הפונקציה mainB1?

כפתרון לשאלה 6 התייחסו לקוד עד הדפסת הקו שמתחיל ב- 1
כפתרון לשאלה 7 התייחסו לקוד בין הדפסות הקווים 1 ל- 2
כפתרון לשאלה 8 התייחסו לקוד בין הדפסות הקווים 2 ל- 3
כפתרון לשאלה 9 התייחסו לקוד בין הדפסות הקווים 3 ל- 4
כפתרון לשאלה 10 התייחסו לקוד לאחר הדפסת הקו שמתחיל ב- 4

שאלה 11:

מה יקרה בעקבות הרצת הפונקציה mainB2?

שאלה 12:

מה יקרה בעקבות הרצת הפונקציה mainB3?

מבחן מועד Y – פתרון

במבחן 12 שאלות. משקל כל שאלה הוא 10 נקודות.

הציון יהיה על 10 השאלות עם הציון הגבוה ביותר.

עבור כל שאלה יש להתייחס לדברים הבאים:

- ✓ במידה והיא מתקמפלת יש לכתוב מהו הפלט, כולל מה שיודפס לאחר הסוגריים המסולסלים שסוגרים את התוכנית
- ✓ במידה והשאלה אינה מתקמפלת, יש לכתוב מדוע
- ✓ במידה וישנה תעופה מהקוד, יש לכתוב מהו הפלט עד התעופה ולהסביר מדוע הקוד עף
- ✓ במידה ויש קוד שלא כל הזיכרון שוחרר, יש לציין זאת

<u>שאלות 1 עד 6 יתייחסו לקטע הקוד הבא:</u>

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

template<class T>
class Singleton
{
protected:
	Singleton() {} // what would happen if instead {} we wrote ;

	Singleton(const Singleton& other);
	const Singleton& operator=(const Singleton& other);

	static T theSingleton;

public:
	static T& getInstance() { return theSingleton; }
};

template<class T>
T Singleton<T>::theSingleton;

class Student
{
	char name[10];
	int id;

public:
	Student(const char* name, int id)
	{
		strcpy(this->name, name);
		this->id = id;
	}
};
```

```
class College : public Singleton<College>
{
	Student* allStudents[10];
	int numOfStudents;

	College(const College& other) = delete;
	College() = default;

	friend class Singleton<College>; // what would happen without this line?

public:
	bool addStudent(Student& theStudent)
	{
		if (numOfStudents == 10)
			return false;

		allStudents[numOfStudents++] = &theStudent;
		return true;
	}

	operator int() const { return numOfStudents; }
};

int main()
{
	College& c1 = Singleton<College>::getInstance();
	College& c2 = Singleton<College>::getInstance();

	Student s1("gogo", 111);
	Student s2("momo", 222);
	c1.addStudent(s1);
	c1.addStudent(s2);

	Student s3("koko", 333);
	c2.addStudent(s3);

	cout << c1 << endl;
	cout << c2 << endl;
}
```

שאלה 1:

מה יקרה בעקבות הרצת ה- main?

פתרון:

הפלט יהיה:
3
3

שאלה 2:

נחליף כעת ב- main הנ"ל את הגדרת המשתנים c1 ו- c2 עם ההגדרה הבאה:

```
College c1 = Singleton<College>::getInstance();
College c2 = Singleton<College>::getInstance();
```

מה יקרה כעת בעקבות הרצת ה- main?

פתרון:

הקוד לא יתקמפל מאחר ויהיה ניסיון ליצר את c1 ואת c2 באמצעות copy c'tor, שהוא נמצא בהרשאת private.

שאלה 3:

נחליף כעת ב- main הנ"ל את הגדרת המשתנים c1 ו- c2 עם ההגדרה הבאה:

```
College c1;
College c2;
```

מה יקרה כעת בעקבות הרצת ה- main?

פתרון:

הקוד לא יתקמפל מאחר ול- College אין default c'tor בהרשאת public

שאלה 4:

במחלקה Singleton הקונסטרקטור מוגדר כך:

```
Singleton() {}
```

מאחר והמחלקה Singleton הינה מחלקת template, מישהו הציע להחליף את השורה כך שתראה כך, ללא מימוש בהמשך:

```
Singleton();
```

מה יקרה כעת בעקבות הרצת ה- main המקורי?

פתרון:

הקוד לא יעבור לינקר: מאחר שהקונסטקטור של College קורא לקונסטרקטור ברירת המחדל של Singleton, ומאחר ולאחרון יש רק הצהרה ללא מימוש, הקוד לא יעבור לינקר.

שאלה 5:

המחלקה College נותנת חברות למחלקה Singleton. האם הדבר הכרחי? נמק.

פתרון:

מתן החברות הכרחי מאחר והקונסטרקטור של College נמצא ב- private, וב- Singleton ישנו משתנה סטטי מטיפוס המחלקה College, כלומר ישנו מעבר בקונסטרקטור שלו. ללא מתן החברות לא תהיה גישה לקונסטרקטור ברירת המחדל בשל הרשאתו.

שאלות 6 עד 12 יתייחסו לקטע הקוד הבא:

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

class Ink
{
	bool isBlack;
	char* manufacturer;

	const Ink& operator=(const Ink& other);
public:
	Ink(const char* manufacturer, bool isBlack)
	{
		this->manufacturer = strdup(manufacturer);
		this->isBlack = isBlack;
		cout << "Creating ink: " << *this << endl;
	}

	Ink(const Ink& other)
	{
		manufacturer = strdup(other.manufacturer);
		isBlack = other.isBlack;
		cout << "Copying ink: " << *this << endl;
	}

	Ink(Ink&& other) : manufacturer(nullptr)
	{
		swap(manufacturer, other.manufacturer);
		isBlack = other.isBlack;
		cout << "Moving ink: " << *this << endl;
	}

	~Ink()
	{
		cout << "Deleting ink: " << *this << endl;
		delete[]manufacturer;
	}

	friend ostream& operator<<(ostream& os, const Ink& i)
	{
		if (i.manufacturer)
			os << i.manufacturer;
		else
			os << "Manufacturer already deleted";

		os << ", " << (i.isBlack ? "black" : "color") << " ink";
		return os;
	}

	const char* getManufacturer() const { return manufacturer; }
};
```

```
class Printer
{
	Ink theInk;
	char* manufacturer;
	bool isScanner;
	int serialNumber;

	static int serialNumberGenerator;

	const Printer& operator=(const Printer& other);
public:
	Printer(const char* manufacturer, const Ink& theInk, bool isScanner = true)
	                                                          : theInk(theInk)
	{
		this->manufacturer = strdup(manufacturer);
		this->isScanner = isScanner;
		serialNumber = ++serialNumberGenerator;
		cout << "Creating Printer: " << *this << endl;
	}

	Printer(const char* manufacturer, Ink&& theInk, bool isScanner = true)
	                                          : theInk(move(theInk))
	{
		this->manufacturer = strdup(manufacturer);
		this->isScanner = isScanner;
		serialNumber = ++serialNumberGenerator;
		cout << "Creating Printer with move ink: " << *this << endl;
	}

	Printer(const Printer& other) : theInk(other.theInk)
	{
		manufacturer = strdup(other.manufacturer);
		isScanner = other.isScanner;
		serialNumber = ++serialNumberGenerator;
		cout << "Copying Printer: " << *this << endl;
	}

	Printer(Printer&& other) : theInk(move(other.theInk))
	{
		swap(manufacturer, other.manufacturer);
		isScanner = other.isScanner;
		serialNumber = serialNumberGenerator;
		cout << "Moving Printer: " << *this << endl;
	}

	~Printer()
	{
		cout << "Deleting Printer: " << *this << endl;
		delete[]manufacturer;
	}

	friend ostream& operator<<(ostream& os, const Printer& p)
	{
		os <<  p.manufacturer
			<< ", S/N: " << p.serialNumber
			<< ", " << (p.isScanner ? "also" : "not") << " a scanner. "
			<< "Ink: " << p.theInk
			<< (strcmp(p.theInk.getManufacturer(), p.manufacturer) == 0 ?
                                                  " (original ink)" : "");
		return os;
	}
};
```

```
int Printer::serialNumberGenerator = 1000;

Printer createPrinterWithOrigInk(const char* manufacturer, bool isBlackInk,
                                 bool isScanner)
{
       return Printer(manufacturer, Ink(manufacturer, isBlackInk), isScanner);
}

void mainB1()
{
       Ink i1("hp", true);
       Printer p1("hp", i1);
       cout << "1 ----------------------\n";

       Printer p2("epson", Ink("hp", true), false);
       cout << "2 ----------------------\n";

       Printer p3(p1);
       cout << "3 ----------------------\n";

       cout << createPrinterWithOrigInk("epson", false, true) << endl;
       cout << "4 ----------------------\n";
}

void mainB2()
{
       Ink i1("hp", true);
       Ink i2("epson", false);

       i1 = i2;
}

void mainB3()
{
       Ink i1("hp", true);
       Printer p1("hp", i1);
       Printer p2("epson", Ink("hp", true), false);

       p1 = p2;
}
```

שאלות 6 – 10:

מה יקרה בעקבות הרצת הפונקציה mainB1?

כפתרון לשאלה 6 התייחסו לקוד עד הדפסת הקו שמתחיל ב- 1

פתרון:

```
Creating ink: hp, black ink
Copying ink: hp, black ink
Creating Printer: hp, S/N: 1001, also a scanner. Ink: hp, black ink (original ink)
```

כפתרון לשאלה 7 התייחסו לקוד בין הדפסות הקווים 1 ל- 2

פתרון:

```
Creating ink: hp, black ink
Moving ink: hp, black ink
Creating Printer with move ink: epson, S/N: 1002, not a scanner. Ink: hp, black
ink
Deleting ink: Manufacturer already deleted, black ink
```

כפתרון לשאלה 8 התייחסו לקוד בין הדפסות הקווים 2 ל- 3

פתרון:

```
Copying ink: hp, black ink
Copying Printer: hp, S/N: 1003, also a scanner. Ink: hp, black ink (original ink)
```

כפתרון לשאלה 9 התייחסו לקוד בין הדפסות הקווים 3 ל- 4

פתרון:

```
Creating ink: epson, color ink
Moving ink: epson, color ink
Creating Printer with move ink: epson, S/N: 1004, also a scanner. Ink: epson,
color ink (original ink)
Deleting ink: Manufacturer already deleted, color ink
epson, S/N: 1004, also a scanner. Ink: epson, color ink (original ink)
Deleting Printer: epson, S/N: 1004, also a scanner. Ink: epson, color ink
(original ink)
Deleting ink: epson, color ink
```

כפתרון לשאלה 10 התייחסו לקוד לאחר הדפסת הקו שמתחיל ב- 4

פתרון:

```
Deleting Printer: hp, S/N: 1003, also a scanner. Ink: hp, black ink (original ink)
Deleting ink: hp, black ink
Deleting Printer: epson, S/N: 1002, not a scanner. Ink: hp, black ink
Deleting ink: hp, black ink
Deleting Printer: hp, S/N: 1001, also a scanner. Ink: hp, black ink (original ink)
Deleting ink: hp, black ink
```

שאלה 11:

מה יקרה בעקבות הרצת הפונקציה mainB2?

פתרון:

הקוד אינו עובר קומפילציה מאחר ואופרטור= של Ink נמצא ב- private

שאלה 12:

מה יקרה בעקבות הרצת הפונקציה mainB3?

פתרון:

הקוד אינו עובר קומפילציה מאחר ואופרטור= של Printer נמצא ב- private