Benjamin I. Czaczkes PhD
The Hebrew University of Jerusalem,
Programming Instruction Unit,
Givat Ram, Jerusalem 91904, Israel

ד"ר בנימין צ'צ'קס
האוניברסיטה העברית בירושלים,
היחידה להוראת התכנות,
גבעת רם, ירושלים 91904

Site:pl1.mscc.huji.ac.il :אתר e-mail: msbc@mscc.huji.ac.il :דואר אלקט' Fax: 972-2-6586246 :פקס Tel: 972-2-6586297 :טל'

# 76641 Fortran and C++ תכנות

מבחן בקורס C++ לפיזיקאים

שמות המורים: יאיר הפנר, בני צ'צ'קס, גדי קסיר
מועד ב' תשס"ב
כל חומר כתוב מותר
משך הזמן: שעתיים

משפחת המצולעים הסגורים מוגדרת כמחלקה בעלת המשתנים הבאים:
int N – מספר הצלעות.
long *SIDES – מערך בגודל N המכיל את אורכי צלעות המצולע.
במחלקה מוגדרת גם פונקציה לחישוב היקף המצולע.
ה CONSTRUCTOR של המחלקה יקבל ערך עבור N ויבנה את המערך SIDES.

ממחלקה זו נגזרו בירושה המחלקות הבאות: מלבנים ומשולשים.
מלבן מוגדר ע"י אורכי שתי צלעות סמוכות ומשולש ע"י אורכי שלוש הצלעות.
ה CONSTRUCTOR של מחלקת המלבנים יקבל שני ערכים עבור שתי צלעות סמוכות של המלבן ויציב את ערכי כל ארבעת הצלעות במערך SIDES.
ה CONSTRUCTOR של מחלקת המשולשים יקבל שלושה ערכים עבור שלושת צלעות המשולש ויציב את ערכי הצלעות במערך SIDES.

1. הגדירו את שלוש המחלקות לעיל.

2. הגדירו CONSTRUCTORS לכל מחלקות.
   שימו לב! את יצירת מערך הצלעות עליכם לכתוב פעם אחת בלבד.

3. הגדירו את פונקצית ההיקף במחלקת הבסיס.

4. שני מצולעים יוגדרו כזהים אם:
   - כל צלע במצולע ראשון נמצאת גם במצולע השני ולהיפך כל צלע במצולע השני נמצאת גם במצולע הראשון (הסדר אינו מחייב).
   - מספר הצלעות של המצולעים שווה.
   - סכום הצלעות של המצולעים שווה.

   לדוגמא המצולע A(3,3,2,1,5) אינו זהה למצולע B(2,2,3,1,5) אך זהה למצולע C(3,2,3,5,1).
   הגדירו את האופרטור == בהתאם במחלקת הבסיס.

הערה:
במידה וקטע תוכנית מסוים נכתב מספר פעמים יש לכתבו כפונקציה.
הנכם רשאים להוסיף כל פונקציה נוספת שתבחרו.

**בהצלחה**

```
/***********************************
  File: E:\Doc\cppf902003\Src\metzula\main.cpp
  Revision By: Benjamin Czaczkes PhD
  Revised on 26/05/2003 22:32:47
 ***********************************/

#include <iostream>

class poly
{
protected:
     int n;
     long * sides;
public:
     poly(int N);
     ~poly(void){delete [] sides;cout<<"poly gone"<<endl;}
     long circ(void) const;
     bool operator==(const poly & rhs);
};

class tri:public poly
{
public:
     tri(long p1, long p2, long p3);
     ~tri(void){cout<<"tri gone"<<endl;}
};

class rect:public poly
{
public:
     rect(long p1, long p2);
     ~rect(void){cout<<"rect gone"<<endl;}
};

main()
{
     return 0;
}

poly::poly(int N)
{
     n=N; sides=new long[n]; cout <<"poly created"<<endl;
}
```

```
long poly::circ(void) const
{
	long t=0;
	for (int i=0;i<n;t+=sides[i++]);
	return t;
}

bool poly::operator==(const poly & rhs)
{
	if (n!=rhs.n) return false;
	if (circ()!=rhs.circ()) return false;
	// make a copy of rhs.sides
	long * t = new long[n];
	int m=0;
	for(int i=0;i<n;i++) t[i]=rhs.sides[i];
	for(int j=0;j<n;j++)
	{
		for(i=0;i<n;i++)
		{
			if (sides[j]==t[i])
			{
				t[i]=-1;
				m++;
				break;
			}
		}
	}
	delete [] t;
	return m==n;
}

tri::tri(long p1, long p2, long p3):poly(3)
{
	sides[0]=p1;sides[1]=p2;sides[2]=p3;
	cout<<"tri created"<< endl;
}

rect::rect(long p1, long p2):poly(4)
{
	sides[0]=sides[2]=p1;
	sides[1]=sides[3]=p2;
	cout<<"rect created"<<endl;
}
```