# פרוייקט בקומפילציה | חלק 2

204287932 - איל ביסמוט - 308200203 | נתנאל סודאי - 31600487 | אדיר אנג'ל

#### <u>: דוגמאות הרצה</u>

1. קיימת פונקציה MAIN בקוד והיא יחידה:

```
func foo() return int
{
    return 1;
}
proc Main(){
}
proc Main(){

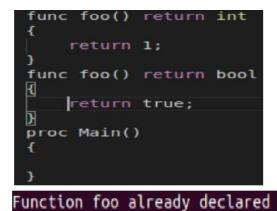
Where is your main?!?!?!
```

2. MAIN לא מקבל ארגומנטים:

```
proc Main(x:int)
{

}
main can not be with parameters
```

3. לא קיימות שתי פונקציות או פרוצדורות עם אותו שם באותו



# : scope לא קיימים שני משתנים עם אותו שם באותו 4

```
func foo(x : bool) return int

var x: int;

return x;

proc Main()
{

Variable 'x' already declared
```

```
func foo() return int
{
    var x: int;
    var x: bool;
    return x;
}

proc Main()
{
}
Variable 'x' already declared
```

## 5. פונקציות ופרוצדורות הוגדרו לפני שמפעילים אותן:

```
proc Main()
{
    foo(true);
}
func foo(x : bool) return int
{
    var x: int;
    return x;
}
Function 'foo' not declared
```

```
proc Main()
{
    foo();
}
proc foo()
{
    var x: int;
}
Function 'foo' not declared
```

#### 6. משתנים הוגדרו לפני שמשתמשים בהם :

```
proc Main()
{

func foo() return int
{
    return x;
}

Variable 'x' not declared
```

7. מספר הארגומנטים האקטואליים שווה למספר הארגומנטים הפורמליים של הפונקציה/פרוצדורה - כמות הארגומנטים בקריאה לפונקציה/פרוצדורה צריכה להיות שווה לכמות הארגומנטים בהגדרת הפונקציה/פרוצדורה:

```
func foo(x,y,z:int) return int
                                           func foo(x,y,z:int) return int
                                               return x;
    return x;
                                           proc Main()
proc Main()
                                              var x : int;
   var x : int;
                                              x = foo(1,2,3);
   x = foo(1,2,3,4);
wrong number or types of argumentsn
in function call (in the scope of 'Main')
```

8. טיפוסים של הארגומנטים בקריאה לפונקציה/פרוצדורה תואמים לטיפוסים בהגדרת הפונקציה/פרוצדורה:

```
func foo(x,y:int;z:real) return int
                                                func foo(x,y:int) return int
                                                    return x;
                                                proc Main()
                                                   var x : int;
                                                   x = foo([1,2]);
   x = foo(1,2,'x');
wrong number or types of argumentsn
in function call (in the scope of 'Main')
```

9. טיפוס הערך המוחזר מהפונקציה תואם לטיפוס ההחזרה המוכרז בכותרת של הפונקציה. וטיפוס ההחזרה של הפונקציה לא יכול להיות מחרוזת:

```
func foo() return string
    return "sss";
proc Main()
   var x : int;
Error: syntax error at line 1
Parser does not expect
```

```
func foo() return int
₹
      var x : char;
      return x;
proc Main()
    var x : int;
Return 'char' instead of 'int', check your 'foo'
```

return x;

var x : int;

proc Main()

# 10. טיפוס הערך המוחזר מהפונקציה תואם לטיפוס המשתנה שלתוכו נכנס הערך שמוחזר מהפונקציה:

```
func foo() return char
{
    var x : char;
    return x;
}
proc Main()
{
    var x : int;
    x = foo();
}
Operator '=' can only be used with identical types
```

```
func foo() return char
{
    var x : char;
    return x;
}
proc Main()
{
    var x : char;
    x = foo();
}
```

#### : bool טיפוס התנאי ב if-הוא מטיפוס.11

```
func foo() return char
{
    var x: int;
    if(x)
    {
        }
        return 'c';
}
proc Main()
{

If condition is 'int' instead of ' bool ', check your 'foo'
```

```
func foo() return char
{
    var x: bool;
    if(x)
    {
        return 'c';
}
proc Main()
```

#### : bool טיפוס התנאי ב-while הוא מטיפוס.12

```
func foo() return char
{
    var x: int;
    while(x)
    {
        }
        return 'c';
}
proc Main()
{
    }
WHILE condition is 'int' instead of ' bool ', check your 'foo'
```

```
func foo() return char

var x: bool;

while(x)
{

   return 'c';
}
proc Main()
{
}
```

: int טיפוס הביטוי המופיע כאינדקס ב- [ ] של מחרוזת הוא מטיפוס

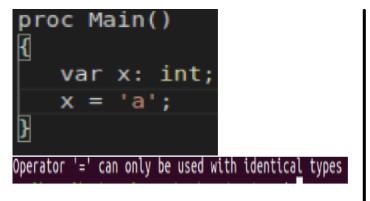
```
proc Main()
{
    var x: string[100];
    x[5.5] = 'a';
}
Index must be int
```

```
proc Main()
{
    var x: string[100];
    x[5] = 'a';
}
OK
```

14. לא משתמשים באופרטור [ ] בשום טיפוס חוץ ממחרוזת:

```
proc Main()
{
    var x: int;
    x[5] = 2;
}
Index operator must be only used with string variables
```

15. טיפוס של המשתנה מצד שמאל של האופרטור השמה ( = ) תואם לטיפוס הביטוי מצד ימין. לשים לב שלתאים של המחרוזת מותר להכניס רק תווים ו null-יכול להיות רק מטיפוס מצביע:



```
proc Main()
{
    var x: string[[20]];
    x = "sssa";
}
```

```
proc Main()
{
    var x: int;
    x = null;
}

NULL can be assigned only
to int* or char* or real* variabels
```

```
proc Main()
{
   var x: int*;
   x = null;
}
```

16. טיפוסים בביטויים (expressions) תואמים:

```
proc Main()
{
   var x: char;
   x = 'a' + 'b';
}
The operator '+' only compatible with int or real
```

```
proc Main()

var x: char;
x = 'a' * 'b';

The operator '*' only compatible with int or real
```

```
proc Main()
{
   var x: bool;
   x = true && false || true;
}
```

ΟK

expression not boolean

```
proc Main()
{
    if ('x' < 't'){
    }
}
The operator '<' only compatible with int or real</pre>
```

```
i>,=>,<,=< ...
proc Main()
{
    if (|2 <= 3 |) {
        }
    }
}</pre>
```

OK

<u>:=! ,==</u> •

```
proc Main()
{
    var x : bool;
    x = 3 == 'a';
}
The operator '==' only compatible with
int,int*,char,char*,real,real* or boolean
```

```
proc Main()
{
    var x : bool;
    x = 3 != 6;
}
```

## int אופרטור ערך מוחלט ( | | ) יכול להיות מופעל על מחרוזות והתוצאה היא

```
proc Main()
{
    var x :int;
    var y : string[100];
    x = |y|;
}

OK

ABS only compatible with int or string
```

<u>:!</u> •

```
proc Main()
{
    var x,y :bool;
    x = !y;
}

OK

proc Main()

var x : bool;
var y : int;
x = !y;
}
operator '!' is only compatible with boolean
```

17. אופרטור & מופעל רק על משתנים מטיפוס int ,real,char או string[i] אוווים מופעל רק על משתנים מטיפוס

```
proc Main()
{
   var x : bool;
   var y : int*;
   y = &x|;
}

operator '&' is only compatible with
int,string[i],real or char
proc Main()
{
   var x : int;
   var y : int*;
   y = &x;
}

OK
```

# 18. אופרטור אונרי ^ מופעל רק על מצביעים:

```
proc Main()
{
    var x : int;
    var y : int*;
    ^y = x;
}
```



```
proc Main()
{
    var x : int;
    var y : int*;
    y = ^x;
}
```

operator '^' is only compatible with pointe<u>r</u>s

### <u>דוגמא חוקית מסכמת:</u>

OK