

**Звіт по лабораторній роботі №1  
з архітектури обчислювальних систем  
студента групи К22  
Ламзіна Олега**

1. В якості мови програмування для тестування швидкодії обч. системи було обрано мову програмування Go 1.5.
2. Крім того я написав скрипт на мові програмування Python, за допомогою якого генерується код на мові програмування Go, шляхом підстановки відповідних типів, значень та операцій в заготовку на Go.
3. Тести проводилися на двох обчислювальних системах:
  - Windows8 64bit, i3-3220 3.3GHz
  - Android 4.1.2, IdeaTab A1000L-F, Dual-core 1.2 GHz Cortex-A9

### Лістинг скрипту:

```
code_generator.py  ✕
1  types          = ["int8", "int16", "int32", "int64", "float32", "float64"]
2  operations     = [["+", "addition"], ["-", "subtract"], ["*", "multiplication"], ["/", "division"]]
3  source_file    = open("test.go", "w+")
4
5  init = {
6      "int8" : "91, 23, 65, 55",
7      "int16" : "10123, 9965, 4532, 1235",
8      "int32" : "1073752832, 1073982652, 1065752, 45648989",
9      "int64" : "6341068276411411200, 668276411411200, 634106456465964100, 4644848",
10
11      "float32" : "464566.64654, 999566.685465, 465.45644978, 4599.99998",
12      "float64" : "164545.11164645, 4641010566.101064654, 464566999.64659994, 4464564566.64654456456"
13  }
14
15
```

```
19 #####
20 template_file = open("template_main.go", "r+")
21 template      = ""
22
23 for line in template_file:
24     template += line
25
26 source_file.write(template)
27
```

```
33 template_file = open("template_run_func.go", "r+")
34 template      = ""
35
36 for line in template_file:
37     template += line
38
39 for t in types:
40     template_new = template.replace("#TYPE", t)
41
42     for i in range(4):
43         template_new = template_new.replace("#OPERATION_%d" % (i + 1), operations[i][0])
44         template_new = template_new.replace("#OPERATION_NAME_%d" % (i + 1), operations[i][1])
45
46     source_file.write(template_new)
47
```

```

53
54 template_file = open("template_func.go", "r+")
55 template      = ""
56
57 for line in template_file:
58     template += line
59
60 for t in types:
61     for op in operations:
62         template_new = template.replace("#TYPE", t)
63         template_new = template_new.replace("#OPERATION_NAME", op[1])
64         template_new = template_new.replace("#OPERATION", op[0])
65         template_new = template_new.replace("#INITIALISE_VARIABLES", init[t])
66
67
68
69     source_file.write(template_new)
70

```

“code\_generator.py” створює “test.go” з файлів “template\_func.go”,  
“template\_main.go” & “template\_run\_func.go”

“template\_main.go” лістинг:

```

1  package main
2
3
4  import "fmt"
5  import "time"
6
7
8  func main(){
9
10     test_run_int8()
11     test_run_int16()
12     test_run_int32()
13     test_run_int64()
14
15     test_run_float32()
16     test_run_float64()
17
18 }
19 |
20
21 func string_linear(x float64) string{
22     result := ""
23     for i := 0; i < int(x); i++){
24         result += "*"
25     }
26
27     return result
28 }

```

“template\_run\_func.go” лістинг:

```
1
2 func test_run_#TYPE() {
3
4     t_1 := test_#TYPE_#OPERATION_NAME_1()
5     t_2 := test_#TYPE_#OPERATION_NAME_2()
6     t_3 := test_#TYPE_#OPERATION_NAME_3()
7     t_4 := test_#TYPE_#OPERATION_NAME_4()
8
9
10    fmt.Printf("%s | %8s | %8.3fM | %32s | %8.3f%%\n",
11        "#OPERATION_1", "#TYPE", 1 / t_1 * 10.0,
12        string_linear(t_1 * 25 / t_1), t_1 * 100 / t_1)
13
14    fmt.Printf("%s | %8s | %8.3fM | %32s | %8.3f%%\n",
15        "#OPERATION_2", "#TYPE", 1 / t_2 * 10.0,
16        string_linear(t_1 * 25 / t_2), t_1 * 100 / t_2)
17
18    fmt.Printf("%s | %8s | %8.3fM | %32s | %8.3f%%\n",
19        "#OPERATION_3", "#TYPE", 1 / t_3 * 10.0,
20        string_linear(t_1 * 25 / t_3), t_1 * 100 / t_3)
21
22    fmt.Printf("%s | %8s | %8.3fM | %32s | %8.3f%%\n",
23        "#OPERATION_4", "#TYPE", 1 / t_4 * 10.0,
24        string_linear(t_1 * 25 / t_4), t_1 * 100 / t_4)
25    fmt.Printf("\n")
26
27 }
28
29
```

## “template\_func.go” лістинг:

```
1
2 func test_#TYPE_#OPERATION_NAME() float64 {
3     var a, b, c, d #TYPE = #INITIALISE_VARIABLES
4
5
6     begin_1 := time.Now()
7     for i := 0; i < 10000000; i++ {
8         b = a
9         a = d
10        c = b
11        d = a
12
13        b = a
14        a = c
15        c = b
16        d = a
17
18        b = a
19        a = c
20        c = b
21        d = a
22
23        b = a
24        a = c
25        c = b
26        d = a
27
28        b = a
29        a = c
30        c = b
31        d = a
32
33        b = a
34        a = c
35        c = b
36        d = a
37
38        b = a
39        a = c
40        c = b
41        d = a
42
43        b = a
44        a = c
45        c = b
46        d = a
47
48        b = a
49        a = c
50        c = b
51        d = a
52
53        b = a
54        a = c
55        c = b
56        d = a
57    }
58    end_1 := time.Since(begin_1);
```

```

61 begin_2 := time.Now()
62 for i := 0; i < 10000000; i++ {
63     d = a #OPERATION b
64     d = b #OPERATION c
65     d = c #OPERATION a
66     d = d #OPERATION a
67
68     d = a #OPERATION b
69     d = b #OPERATION c
70     d = c #OPERATION a
71     d = d #OPERATION a
72
73     d = a #OPERATION b
74     d = b #OPERATION c
75     d = c #OPERATION a
76     d = d #OPERATION a
77
78     d = a #OPERATION b
79     d = b #OPERATION c
80     d = c #OPERATION a
81     d = d #OPERATION a
82
83     d = a #OPERATION b
84     d = b #OPERATION c
85     d = c #OPERATION a
86     d = d #OPERATION a
87
88     d = a #OPERATION b
89     d = b #OPERATION c
90     d = c #OPERATION a
91     d = d #OPERATION a
92
93     d = a #OPERATION b
94     d = b #OPERATION c
95     d = c #OPERATION a
96     d = d #OPERATION a
97
98     d = a #OPERATION b
99     d = b #OPERATION c
100    d = c #OPERATION a
101    d = d #OPERATION a
102
103    d = a #OPERATION b
104    d = b #OPERATION c
105    d = c #OPERATION a
106    d = d #OPERATION a
107
108    d = a #OPERATION b
109    d = b #OPERATION c
110    d = c #OPERATION a
111    d = d #OPERATION a
112
113 }
114 end_2 := time.Since(begin_2)
115
116 a = d
117 d = b
118 b = c
119 c = a
120
121 return end_2.Seconds() - end_1.Seconds()
122 }

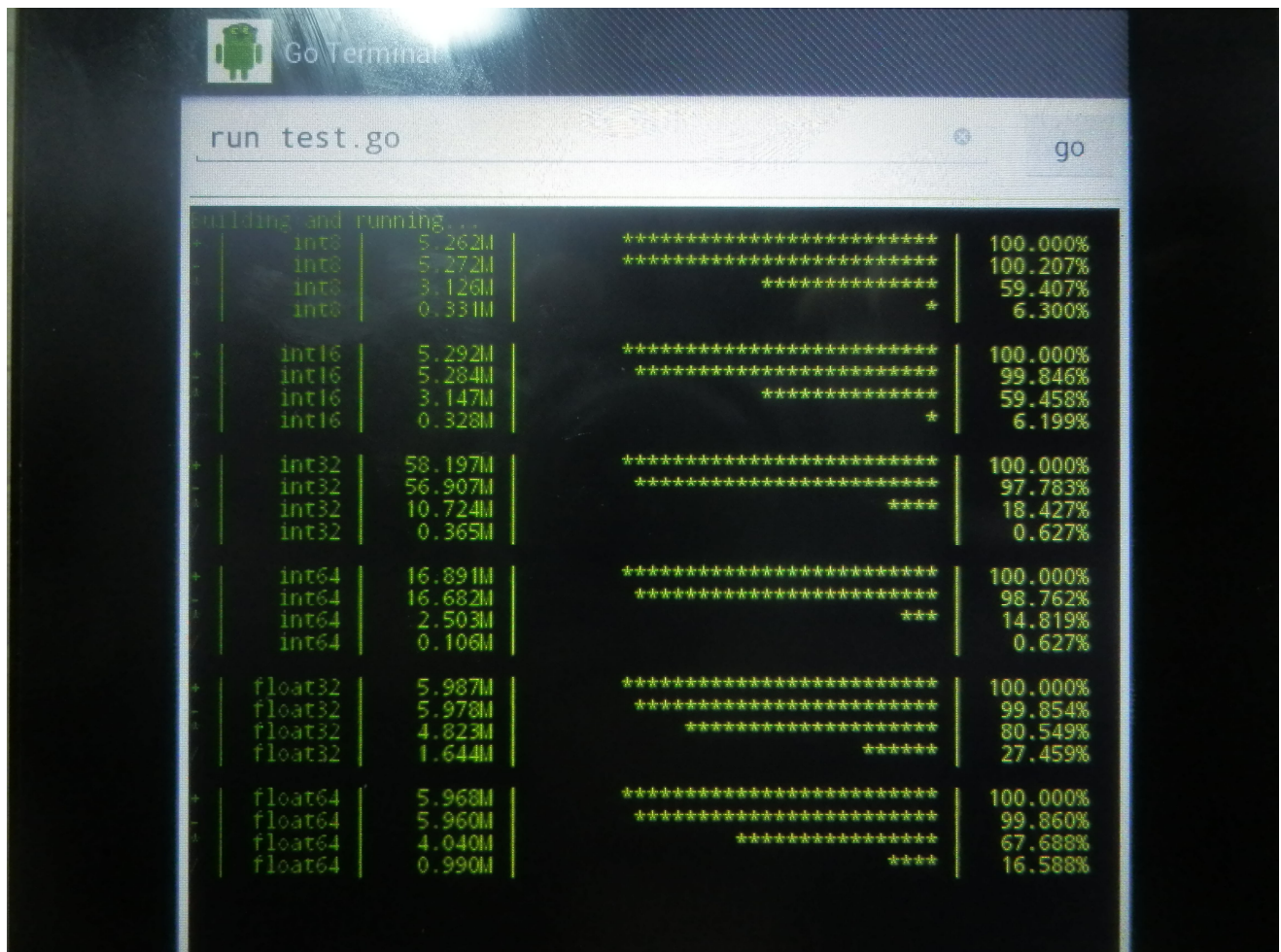
```

## Результати:

Windows8, i3:

+	int8	11.1	10 <sup>9</sup>	*****	100.000%
-	int8	11.8	10 <sup>9</sup>	*****	105.887%
*	int8	4.0	10 <sup>9</sup>	*****	36.000%
/	int8	0.4	10 <sup>9</sup>		3.715%
+	int16	10.8	10 <sup>9</sup>	*****	100.000%
-	int16	12.1	10 <sup>9</sup>	*****	112.115%
*	int16	4.1	10 <sup>9</sup>	*****	37.753%
/	int16	0.4	10 <sup>9</sup>		3.842%
+	int32	11.8	10 <sup>9</sup>	*****	100.000%
-	int32	9.7	10 <sup>9</sup>	*****	82.922%
*	int32	4.0	10 <sup>9</sup>	*****	34.340%
/	int32	0.4	10 <sup>9</sup>		3.501%
+	int64	11.4	10 <sup>9</sup>	*****	100.000%
-	int64	11.1	10 <sup>9</sup>	*****	97.215%
*	int64	4.0	10 <sup>9</sup>	*****	34.999%
/	int64	0.1	10 <sup>9</sup>		1.000%
+	float32	1.0	10 <sup>9</sup>	*****	100.000%
-	float32	1.0	10 <sup>9</sup>	*****	99.479%
*	float32	0.6	10 <sup>9</sup>	*****	61.022%
/	float32	0.2	10 <sup>9</sup>	*****	23.830%
+	float64	5.5	10 <sup>9</sup>	*****	100.000%
-	float64	5.6	10 <sup>9</sup>	*****	102.817%
*	float64	5.6	10 <sup>9</sup>	*****	101.387%
/	float64	0.2	10 <sup>9</sup>	*	4.382%

Android 4.1.2, Dual-core 1.2 GHz Cortex-A9



Go Terminal

run test.go

building and running...

+	int8	5.262M	*****	100.000%
-	int8	5.272M	*****	100.207%
*	int8	3.126M	*****	59.407%
/	int8	0.331M	*	6.300%
+	int16	5.292M	*****	100.000%
-	int16	5.284M	*****	99.846%
*	int16	3.147M	*****	59.458%
/	int16	0.328M	*	6.199%
+	int32	58.197M	*****	100.000%
-	int32	56.907M	*****	97.783%
*	int32	10.724M	****	18.427%
/	int32	0.365M		0.627%
+	int64	16.891M	*****	100.000%
-	int64	16.682M	*****	98.762%
*	int64	2.503M	***	14.819%
/	int64	0.106M		0.627%
+	float32	5.987M	*****	100.000%
-	float32	5.978M	*****	99.854%
*	float32	4.823M	*****	80.549%
/	float32	1.644M	*****	27.459%
+	float64	5.968M	*****	100.000%
-	float64	5.960M	*****	99.860%
*	float64	4.040M	*****	67.688%
/	float64	0.990M	****	16.588%

Слід додати також те що тест веде себе стабільно на різних запусках:

```

D:\Kindle\Kindle Sync\KNU\2 course\Architecture of Computer Systems\Lab1 - Speed
+ | int8 | 11.1 10^9 | ***** | 100.000%
- | int8 | 11.8 10^9 | ***** | 105.887%
* | int8 | 4.0 10^9 | ***** | 36.000%
/ | int8 | 0.4 10^9 | ***** | 3.715%

+ | int16 | 10.8 10^9 | ***** | 100.000%
- | int16 | 12.1 10^9 | ***** | 112.115%
* | int16 | 4.1 10^9 | ***** | 37.753%
/ | int16 | 0.4 10^9 | ***** | 3.842%

+ | int32 | 11.8 10^9 | ***** | 100.000%
- | int32 | 9.7 10^9 | ***** | 82.922%
* | int32 | 4.0 10^9 | ***** | 34.340%
/ | int32 | 0.4 10^9 | ***** | 3.501%

+ | int64 | 11.4 10^9 | ***** | 100.000%
- | int64 | 11.1 10^9 | ***** | 97.215%
* | int64 | 4.0 10^9 | ***** | 34.999%
/ | int64 | 0.1 10^9 | ***** | 1.000%

+ | float32 | 1.0 10^9 | ***** | 100.000%
- | float32 | 1.0 10^9 | ***** | 99.479%
* | float32 | 0.6 10^9 | ***** | 61.022%
/ | float32 | 0.2 10^9 | ***** | 23.830%

+ | float64 | 5.5 10^9 | ***** | 100.000%
- | float64 | 5.6 10^9 | ***** | 102.817%
* | float64 | 5.6 10^9 | ***** | 101.387%
/ | float64 | 0.2 10^9 | ***** | 4.382%

D:\Kindle\Kindle Sync\KNU\2 course\Architecture of Computer Systems\Lab1 - Speed
+ | int8 | 10.8 10^9 | ***** | 100.000%
- | int8 | 11.1 10^9 | ***** | 102.781%
* | int8 | 4.1 10^9 | ***** | 37.756%
/ | int8 | 0.4 10^9 | ***** | 3.787%

+ | int16 | 11.4 10^9 | ***** | 100.000%
- | int16 | 11.4 10^9 | ***** | 99.967%
* | int16 | 4.1 10^9 | ***** | 36.069%
/ | int16 | 0.4 10^9 | ***** | 3.585%

+ | int32 | 11.8 10^9 | ***** | 100.000%
- | int32 | 9.7 10^9 | ***** | 82.938%
* | int32 | 4.1 10^9 | ***** | 35.055%
/ | int32 | 0.4 10^9 | ***** | 3.470%

+ | int64 | 11.1 10^9 | ***** | 100.000%
- | int64 | 10.8 10^9 | ***** | 97.293%
* | int64 | 4.0 10^9 | ***** | 36.363%
/ | int64 | 0.1 10^9 | ***** | 1.006%

+ | float32 | 1.0 10^9 | ***** | 100.000%
- | float32 | 1.0 10^9 | ***** | 98.209%
* | float32 | 0.6 10^9 | ***** | 60.568%
/ | float32 | 0.2 10^9 | ***** | 23.762%

+ | float64 | 5.6 10^9 | ***** | 100.000%
- | float64 | 5.6 10^9 | ***** | 100.005%
* | float64 | 5.6 10^9 | ***** | 98.618%
/ | float64 | 0.2 10^9 | ***** | 4.234%

```