GeonSsembly Lang Version

```
.header
int count
int sum
int i
.code
//count = 10;
LOD 10
STO @count
//sum = 0;
LOD 0
STO @sum
//i=0;
LOD 0
STO @i
start:
//read i
//MBR,AC=Memory[@i]
LOD @i
//AC = AC-count (i-count)
SUB @count
//if(AC=0) then Jump to : end label
EZJ end
//read sum //MBR,AC = Memory[@sum]
LOD @sum
//AC = AC+1 (Sum=Sum+1)
ADD 1
//Memory[@sum] = AC
STO @sum
//i = i+1
LOD @i
ADD 1
STO @i
JMP start
end:
PRT @sum
HALT
.end
```

CODE : 0to10 (간단한 반복문)

```
static main(){
    int count = 10;
    int sum = 0;
    int i = 0 ;

    while((i<count){
        sum = sum + 1;
        i = i + 1;
    }
    System.out.println(sum);
}
```

ObjectCode

```
12  18
0x0010a
0x10300
0x00100
0x10304
0x00100
0x10308
0x10108
0x10600
0x0130f
0x10104
0x00501
0x10304
0x10108
0x00501
0x10308
0x01005
0x15004
0x00000
```

CODE : CallNestedFuncAndParams (함수중첩호출 + 파라매터 전달)

```
static main(){
    int num = 2;
    Student student = new Student();
    int result = student.getAverage(2);
    System.out.println(result);
}

Class Student{
        int kor = 60;
        int eng - 50;
        public int getAverage(int number){
            int sum = this.kor + this.eng;
            int average = calculateAverage(this.sum, number);
            return average;
        }
        public int calculateAverage(int paramSum, int paramNum){
                return (paramSum / paramNum);
        }
}
```

GeonSsembly Lang Version

```
.header
int num
int result
object student
.code
//int num = 2
LOD 2
STO @num
//student.kor variable -> into heap seg
NEWINT student.*kor
student.kor variable = 60
LOD 60
STO student.*kor
//student.eng
NEWINT student.*eng
LOD 50
STO student.*eng
JMP getAverageFunction
:copyToResult
//result = student.getAverage() in main()
LOD #returnValue
STO @result
//getAverage's stack frame(Activation record) remove
STF #returAddress
```

```
PRT @result
HALT
:getAverageFunction
//Dinamic Link
DL #returnAddress
//PUSH Local Variables
PUSHINT #returnValue
PUSHINT #number
PUSHINT #sum
PUSHINT #average
//link parameter with global variable
LOD @num
STO #number
//sum = this.kor+this.eng
LOD student.*kor
ADD student.*eng
STO #sum
//call inner function
JMP calculateAverageFunction
:getAverageReturn
// int average = calculateAverage();
LOD #innerReturnValue
STO #average
STF #innerReturnAddress
//return average
LOD #average
STO #returnValue
//jump to main()
JMP copyToResult
calculateAverageFunction:
DL #innerReturnAddress
PUSHINT #innerReturnValue
PUSHINT #paramSum
PUSHINT #paramNum
LOD #sum
STO #paramSum
LOD #number
STO #paramNum
//return paramSum/paramNum
LOD #paramSum
DIV #paramNum
STO #innerReturnValue
JMP getAverageReturn
.end
```

ObjectCode

```
12 41
0x00102
0x10300
0x33008
0x0013c
0x30308
0x3300c
0x00132
0x3030c
0x0100d
0x20100
0x10304
0x22200
0x15004
0x00000
0x22000
0x22100
0x22108
0x2210c
0x22110
0x30108
0x3050c
0x2030c
0x0101c
0x20114
0x20310
0x22214
0x20110
0x20300
0x01008
0x22014
0x22114
0x2211c
0x22120
0x2010c
0x2031c
0x20108
0x20320
0x2011c
0x20820
0x20314
0x01016
```

CODE : AdvancedLoop (반복 객체, 반복 함수호출)

```
static main(){
    int num = scanner.nextInt();
    int i = 0;
    while(i<num){
            Student student = new Student();
            student.kor = scanner.nextInt();
            student.eng = scanner.nextInt();
            int result = student.getAverage();
            System.out.println(result);
            i++;
    }
}
Class Student{
            int kor;
            int eng;
            public int getAverage(){
                int sum = this.kor + this.eng;
                int average = calculateAverage(this.sum);
                return average;
            }
            public int calculateAverage(int paramSum){
                        return (paramSum / 2);
            }
}
```

GeonSsembly Lang Version

```
.header
int num
int i
int result
object student
.code
//num = scanner.nextInt
INP
STO @num
//i=0
LOD 0
STO @i
loop:
//student.kor variable -> into heap seg
NEWINT student.*kor
//student.kor variable = scanner.nextInt
INP
STO student.*kor
//student.eng into heap seg
NEWINT student.*eng
```

```
INP
STO student.*eng
JMP getAverageFunction
:copyToResult
//result = student.getAverage() in main()
LOD #returnValue
STO @result
//getAverage's stack frame(Activation record) remove
STF #returAddress
PRT @result
//i++
LOD @i
ADD 1
STO @i
//if(i<num) then loop
LOD @i
SUB @num
BZJ loop
HALT
:getAverageFunction
//Dinamic Link
DL #returnAddress
//PUSH Local Variables
PUSHINT #returnValue
PUSHINT #sum
PUSHINT #average
//sum = this.kor+this.eng
LOD student.*kor
ADD student.*eng
STO #sum
//call inner function
JMP calculateAverageFunction
:getAverageReturn
// int average = calculateAverage();
LOD #innerReturnValue
STO #average
STF #innerReturnAddress
//return average
LOD #average
STO #returnValue
//jump to main()
JMP copyToResult
calculateAverageFunction:
DL #innerReturnAddress
PUSHINT #innerReturnValue
PUSHINT #paramSum
LOD #sum
STO #paramSum
//return paramSum/2
LOD #paramSum
DIV 2
STO #innerReturnValue
JMP getAverageReturn
.end
```

ObjectCode

```
16 45
0x06000
0x10300
0x00100
0x10304
0x3300c
0x06000
0x3030c
0x33010
0x06000
0x30310
0x01015
0x20100
0x10308
0x22200
0x15008
0x10104
0x00501
0x10304
0x10104
0x10600
0x01203
0x00000
0x22000
0x22100
0x22108
0x2210c
0x3010c
0x30510
0x20308
0x01023
0x20110
0x2030c
0x22210
0x2010c
0x20300
0x0100a
0x22010
0x22110
0x22118
0x20108
0x20318
0x20118
0x00802
0x20310
```

CODE : Factorial (입력값 num에 대한 팩토리얼 구하기)

GeonSsembly Lang Version

```
.header
int num
int result
.code
INP
STO @num
JMP factorialFunction
:copyToResult
LOD #returnValue
STO @result
STF #returnAddress
PRT @result
HALT
:factorialFunction
DL #returnAddress
PUSHINT #returnValue
PUSHINT #n
LOD @num
STO #n
:recursiveLoop
LOD #n
SUB 1
STO #returnValue
//if(n==1) then return
LOD #n
SUB 1
STO #n
LOD #n
EQ 1
EZJ returnLoop
JMP recursiveLoop
:returnLoop
LOD #returnValue
MUL #n
STO #returnValue
LOD #n
EQ @num
EZJ copyToResult
LOD #n
ADD 1
STO #n
JMP returnLoop
.end
```

ObjectCode

```
8 33
0x06000
0x10300
0x01007
0x20100
0x10304
0x22200
0x15004
0x00000
0x22000
0x22100
0x22108
0x10100
0x20308
0x20108
0x00601
0x20300
0x20108
0x00601
0x20308
0x20108
0x00a01
0x01316
0x0100c
0x20100
0x20708
0x20300
0x20108
0x10a00
0x01302
0x20108
0x00501
0x20308
0x01016
```