204324 ปฏิบัติการระบบคอมพิวเตอร์
เตรียมตัวสอบกลางภาค
1.

```
#include <stdio.h>
int main()
{
   int a = -10;
   int x = 10;
   // printf("a: %d\n", a);
   a = a >> 31;
   // printf("a>>31 %d\n", a);
   int y = a << 1;
   // printf("a<<1 %d\n", y);
   printf("x = %d\n", x - 1 + (a * y));
   return 0;
}
```

2.

```
.data
i: .space 4
A: .space 80

.text
.globl main

main:
   li      $t0, 0      # $t0 = 0  ; i = 0

   li      $t1, 1      # $t1 = 1  ; const 1
   li      $t2, 2      # $t2 = 2  ; const 2
```

```
   la      $s0, A           # load address A[0]
   addi    $t5, $s0, 0      # load array address

loop:
   slti    $t3, $t0, 20     # check i < 20
   beq     $t3, $zero, end

   div     $t0, $t2         # mod 2
   mfhi    $t4              # result of mod

   beq     $t4, $zero, One # go to A[i] = 1

   li      $t7, 0
   sw      $t7, 0($t5)       #A[i] = 0

   li      $v0,1
   lw      $t8,($t5)
   move    $a0,$t8
   syscall

   addi    $t0, $t0, 1      # i++
   addi    $t5, $t5, 4      # go to A[i++]
   j loop

One:
   sw      $t1, 0($t5)      # A[i] = 1

   li      $v0,1
    lw      $t8,($t5)
    move    $a0,$t8
    syscall

   addi    $t0, $t0, 1      # i++
```

```
   addi    $t5, $t5, 4     # go to A[i++]
   j loop

end:
    li $v0,10
    syscall
```

3.

```
.data
   promptMessage:  .asciiz    "Enter a number : "
   resultMessage:  .asciiz    "\nThe result is : "
   theNumber:      .word       0
   theAnswer:      .word       0

.text
   .globl main

   main:
       #Read Input
       li      $v0, 4
       la      $a0, promptMessage
       syscall

       li      $v0, 5
       syscall

       sw      $v0, theNumber

       #Call function
       lw      $a0, theNumber
       jal     numDigit
       sw      $v0, theAnswer
```

```
        #Print
        li      $v0, 4
        la      $a0, resultMessage
        syscall

        li      $v0, 1
        lw      $a0, theAnswer
        syscall

        # End
        li      $v0, 10
        syscall

    .globl numDigit
    numDigit:
        addi    $sp, $sp, -8    #save space for 2 words
        sw      $ra, 4($sp)     #save return address
        sw      $a0, 0($sp)     #save argument

        slti    $t0, $a0, 10    #test n<10
        beq     $t0, $zero, La  #else
        addi    $v0, $zero, 1   #return 1
        addi    $sp, $sp, 8     #pop stack
        jr      $ra

    La:
        div     $a0, $a0, 10    # n/10
        jal     numDigit        # jump to numDigit and save
position to $ra
        lw      $a0, 0($sp)     #restore n
        lw      $ra, 4($sp)     #return address
        addi    $sp, $sp, 8     #pop 2 items
        addi    $v0, $v0, 1     # result = 1 + result
```

```
        jr      $ra             # return
```

4.

Movz If rt = 0 then rd ← rs.

Movn If rt ≠ 0 then rd ← rs.

ที่อาจารย์ไม่ให้ทำน่าจะเกิดจากการที่ ยากที่จะทำไม่ให้เกิด delay แต่อาจารย์อาจเก็บไว้สำหรับ pipelined ไม่ก็ multi cycle mips

5. เราสามารถ นำมาแปลงเป็นโค๊ดได้ง่าย และสามารถแก้ไข Grammar ได้ง่าย นแกจากนั้นยังยืดหยุ่นกว่า BNF เนื่องจากมีการใช้ Regular Expression ด้วย

6. [0-9]* | 0 เมื่อ val หลุดจาก loop จะคืนค่า digit constant กลับไป
เช่น รับ '123' ก็จะนำ (1*10 + 2)*10 + 3 และคืนค่า int 123

7.

$A \rightarrow \text{"}\varepsilon\text{"} \mid B$

$B \rightarrow (A) \mid BA$

จะเห็นว่า

$B = (A) \mid A^{+}$

จาก $A \rightarrow \text{"}\varepsilon\text{"} \mid B$

จะได้

$A = A^{*} \mid (A)$

//เพื่อนสอนมานะครับ โค๊ดอาจเหมือนบ้าง พอทดๆไปมันก็กลับมารูปแบบเดิมที่เพื่อนสอน

```
#include <assert.h>
#include <stdio.h>
#include <stdlib.h>
static FILE *f;
static int ch;
static unsigned int val;
enum
{
   lparen,
   rparen,
   eof,
   illegal
};
```

```
static void SInit(char *filename)
{
   ch = EOF;
   f = fopen(filename, "r+t");
   if (f != NULL)
       ch = getc(f);
}

static int SGet()
{
   register int sym;
   while ((ch != EOF) && (ch <= ' '))
       ch = getc(f);
   switch (ch)
   {
   case EOF:
       sym = eof;
       break;
   case '(':
       sym = lparen;
       ch = getc(f);
       break;
   case ')':
       sym = rparen;
       ch = getc(f);
       break;
   default:
       sym = illegal;
   }
   return sym;
```

```
}

static int sym;
static void A()
{
   while (sym == lparen)
   {
       sym = SGet();
       A();
       assert(sym == rparen);
       sym = SGet();
   }
}

int main(int argc, char *argv[])
{
   register int result;
   if (argc == 2)
   {
       SInit(argv[1]);
       sym = SGet();
       A();
       printf("Accept \n");
   }
   else
   {
       printf("usage: Aeval <filename>\n");
   }
   return 0;
}
```

8. คิดว่าเป็น Pipelined Mips ,Multicycle Mips หรือ Compiler เนื่องจากการทำ Movn Movz บน Single cycle อาจทำให้ช้า อาจารย์เลยยกเลิกออกไป แต่ถ้าเราทำ Pipelined อาจทำให้เร็วขึ้นได้