Introduction

需要用RISC-V實現下列的數學式來計算結果

$$F(x) = \begin{cases} 2 \cdot x + F(\frac{x}{5}), & x > 20 \\ F(x-2) + F(x-3), & 10 < x \leq 20 \\ F(x-1) + F(x-2), & 1 < x \leq 10 \\ 1, \quad x = 0 \\ 5, \quad x = 1 \\ -1, \quad otherwise \end{cases}$$

```
Example:
```

Input:3 我們會得到 output:11

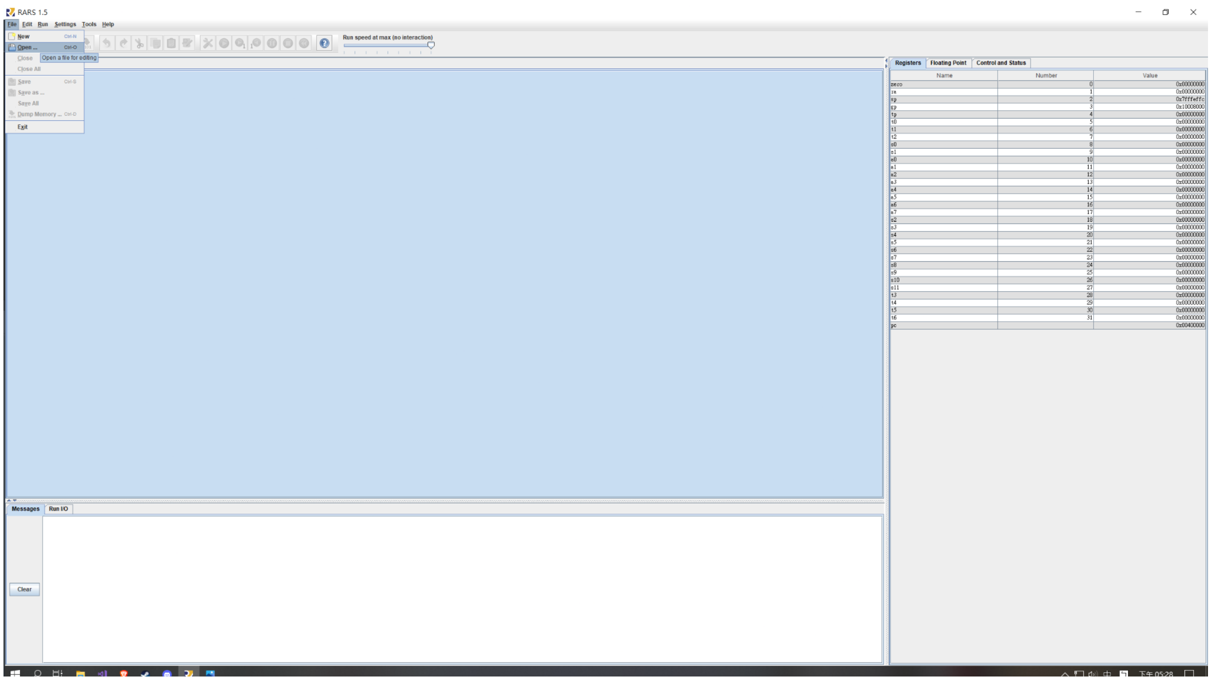
因為F(3)=F(2)+F(1)=F(1)+F(0)+F(1)=11

How to execute

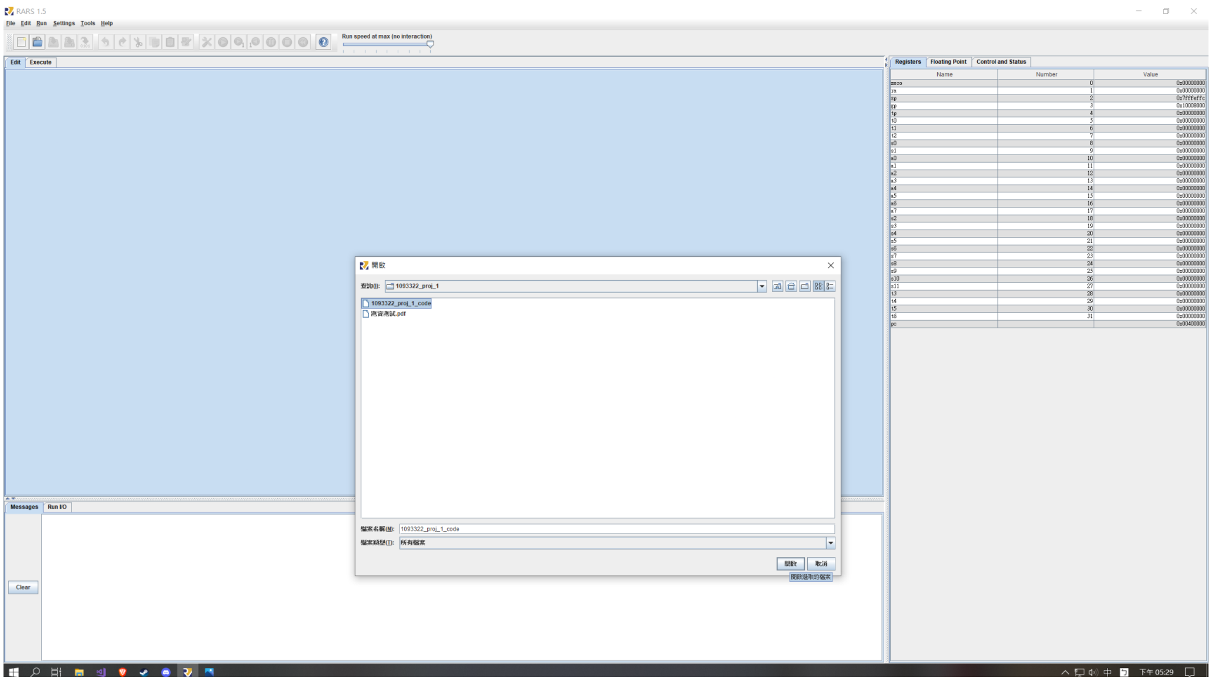
下載正確的Java版本:https://www.java.com/zh-TW/download/

然後下載RARS:https://github.com/TheThirdOne/rars

點選File->open->s1093322_proj_1_code檔



Name	Number	Value
zero	0	0x00000000
ra	1	0x00000000
sp	2	0x7fffeffc
gp	3	0x10008000
tp	4	0x00000000
t0	5	0x00000000
t1	6	0x00000000
t2	7	0x00000000
s0	8	0x00000000
s1	9	0x00000000
a0	10	0x00000000
a1	11	0x00000000
a2	12	0x00000000
a3	13	0x00000000
a4	14	0x00000000
a5	15	0x00000000
a6	16	0x00000000
a7	17	0x00000000
s2	18	0x00000000
s3	19	0x00000000
s4	20	0x00000000
s5	21	0x00000000
s6	22	0x00000000
s7	23	0x00000000
s8	24	0x00000000
s9	25	0x00000000
s10	26	0x00000000
s11	27	0x00000000
t3	28	0x00000000
t4	29	0x00000000
t5	30	0x00000000
t6	31	0x00000000
pc		0x00400000



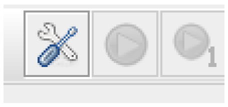
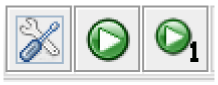
查詢(I): 1093322_proj_1

1093322_proj_1_code
測資測試.pdf

檔案名稱(N): 1093322_proj_1_code
檔案類型(T): 所有檔案

Registers | Floating Point | Control and Status

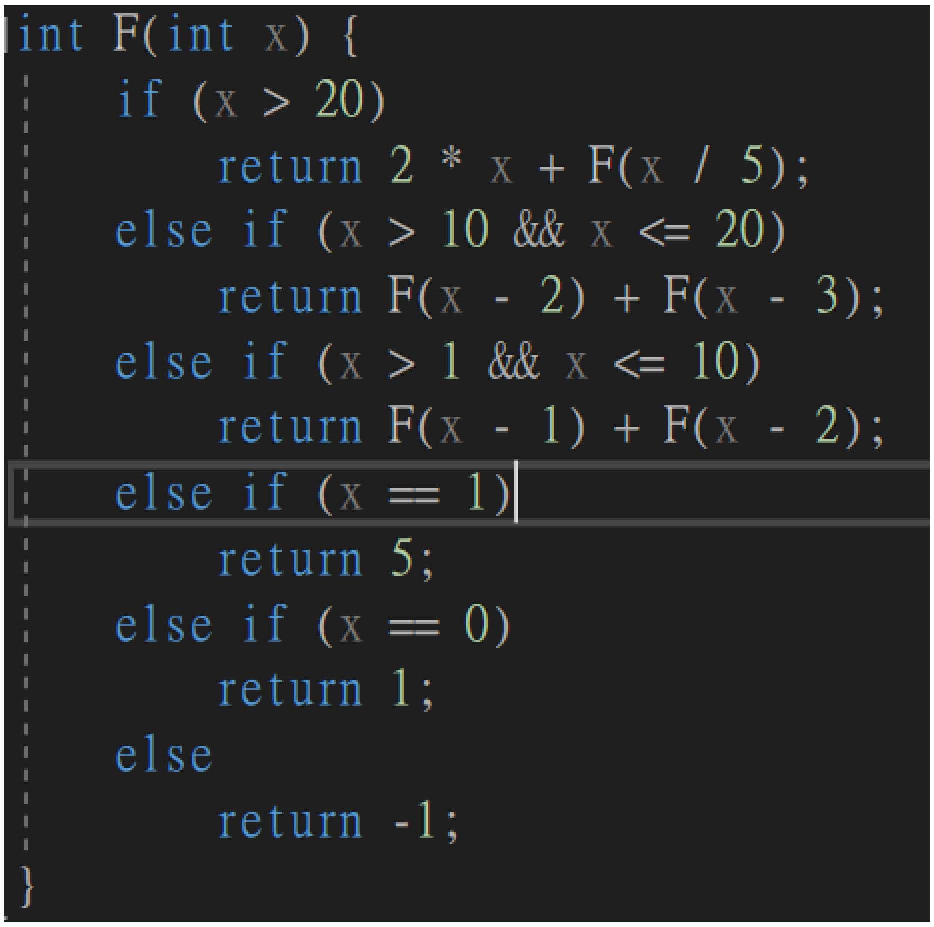
Name	Number	Value
zero	0	0x00000000
ra	1	0x00000000
sp	2	0x7fffeffc
gp	3	0x10008000
tp	4	0x00000000
t0	5	0x00000000
t1	6	0x00000000
t2	7	0x00000000
s0	8	0x00000000
s1	9	0x00000000
a0	10	0x00000000
a1	11	0x00000000
a2	12	0x00000000
a3	13	0x00000000
a4	14	0x00000000
a5	15	0x00000000
a6	16	0x00000000
a7	17	0x00000000
s2	18	0x00000000
s3	19	0x00000000
s4	20	0x00000000
s5	21	0x00000000
s6	22	0x00000000
s7	23	0x00000000
s8	24	0x00000000
s9	25	0x00000000
s10	26	0x00000000
s11	27	0x00000000
t3	28	0x00000000
t4	29	0x00000000
t5	30	0x00000000
t6	31	0x00000000
pc		0x00400000

- 1.請點這顆像工具的按鈕
    - 
- 3.如果編譯沒錯,右邊兩個綠燈會亮起
    - 
    - 第一顆綠燈會全部執行,第二顆綠燈為逐行執行

接著輸入想要的數字就好了

Implement

如果用C++實現的話是這樣:

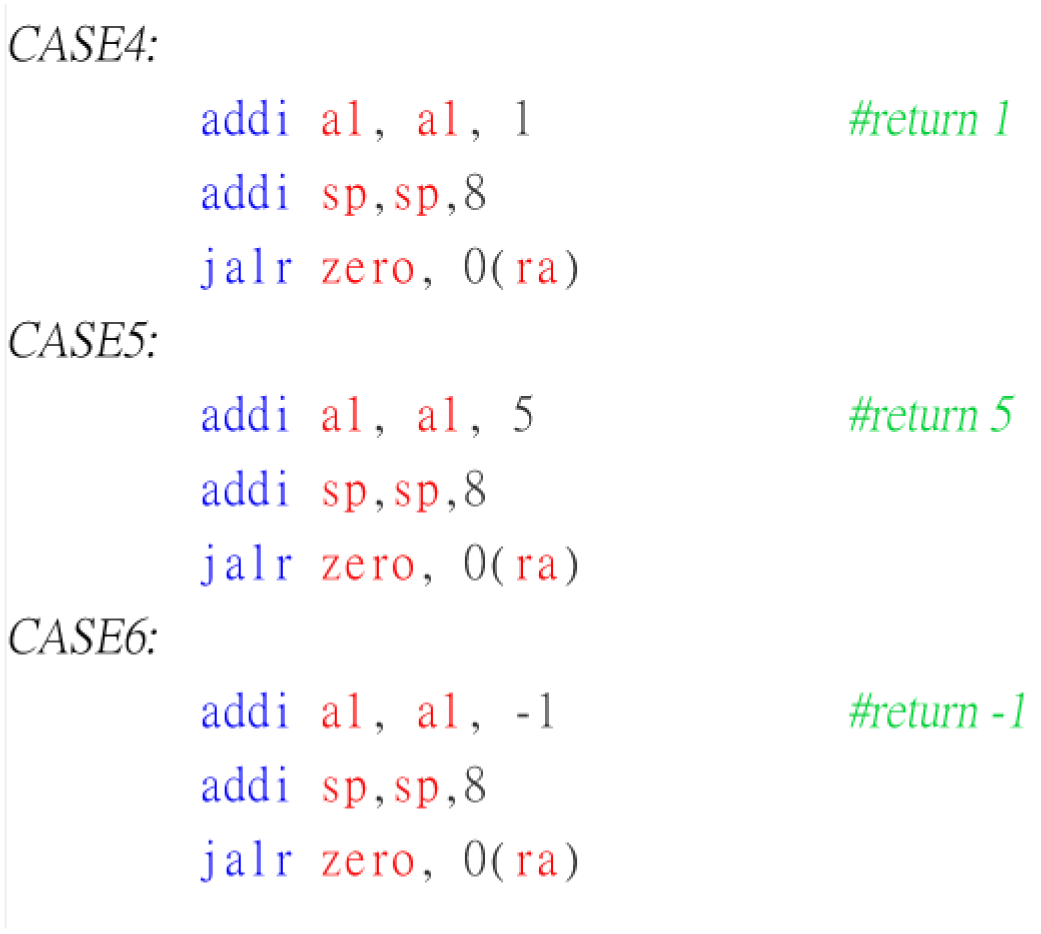


```c
int F(int x) {
    if (x > 20)
        return 2 * x + F(x / 5);
    else if (x > 10 && x <= 20)
        return F(x - 2) + F(x - 3);
    else if (x > 1 && x <= 10)
        return F(x - 1) + F(x - 2);
    else if (x == 1)
        return 5;
    else if (x == 0)
        return 1;
    else
        return -1;
}
```

我先將各個條件式在function裡分類 如下

```
function:                       #a1->total value a0->x value
        addi sp, sp, -8
        sw ra, 4(sp)
        sw a0, 0(sp)
        blt  a0, zero, CASE6          #otherwise
        beq  a0, zero, CASE4          #x==0
        addi t0, zero, 1
        beq  a0, t0, CASE5            #x==1
        addi t0, t0, 9
        ble  a0, t0, CASE3            #x>1 && x<=10
        addi t0, t0, 10
        ble  a0, t0, CASE2            #x>10 && x<=20
        beq  zero , zero, CASE1       #x>20
        addi sp, sp, 8
        jalr zero, 0(ra)              #return
```

再依照不同case來實現輸入在不同情況下所要執行的運算



```
CASE4:
        addi  a1, a1, 1          #return 1
        addi  sp,sp,8
        jalr  zero, 0(ra)
CASE5:
        addi  a1, a1, 5          #return 5
        addi  sp,sp,8
        jalr  zero, 0(ra)
CASE6:
        addi  a1, a1, -1         #return -1
        addi  sp,sp,8
        jalr  zero, 0(ra)
```

CASE4、5、6只需要依照輸入範圍Return 對應的常數值就好

CASE2:

```
        addi  a0, a0, -2
        jal   ra, function          #F(X-2)
        lw ra, 4(sp)
        addi  a0, a0, -1
        jal   ra, function          #F(X-3)
        lw ra, 4(sp)
        lw a0, 0(sp)
        addi  sp, sp, 8
        jalr zero, 0(ra)
```

CASE3:

```
        addi  a0, a0, -1
        jal   ra, function          #F(X-1)
        lw ra, 4(sp)
        addi  a0, a0, -1
        jal   ra, function          #F(X-2)
        lw ra, 4(sp)
        lw a0, 0(sp)
        addi  sp,sp,8
        jalr zero, 0(ra)
```

Case2、3改變x的值後再跑一次fuction實現遞迴，最後加總在a1的值就是答案

CASE1:

```
        slli  a0, a0, 1             #x*=2
        addi  t1, a0, 0             #remove to t1
        div   a0, a0, t3
        jal   ra, function          #F(x/5)
        lw ra, 4(sp)
        lw a0, 0(sp)                #load original x and ra
        add a1, a1, t1              #a1=2*x+F(5/x)
        addi  sp, sp, 8             #pop stack
        jalr zero, 0(ra)
```

Case1先將兩倍的x存起來放在t1，然後除以十(因為前面乘過二)再跑一次function最後得到的a1的值再加上原本存在 t1的兩倍的x的值就能得到答案

```
.data
        endl:     .string"\n"
        Input: .string"Input a number:\n"
        Output: .string"\nThe damage:\n"
.text
main:
        addi t3, zero, 10
        addi t2, zero, 2
        la a0, Input         #Input a num
        li a7, 4
        ecall
        li a7, 5             #a7是5 為ReadInt,值存於a0
        ecall
        jal ra, function
        la a0, Output
        li a7, 4             #output string
        ecall
        addi a0, a1, 0       #store ans in a0
        li a7,1              #output integer
        ecall
        li a7, 10            #exit
        ecall
```

由main來印出字串跟最終結果