```
// 1 esercizio
//scambio
iload A
iload B
istore A
istore B

//2 esercizio
//carico
iload A
iload B
//sottrazione
isub
//verifico che la sottrazione sia < 0
iflt s1
{
        iload A
        iload B
        istore A
        istore B
}
s1

//esercizio 3
bipush 1
istore I
bipush 0
istore S
bipush 32
istore MAX
iload S
iload MAX
ciclo: ificmpeq S1
        inc I 1
        iload I
        iload S
        iadd
        istore S
        goto ciclo
S1 HALT

//esercizio 6
//carico
LDC_W OBJFER
ILOAD A
ILOAD B
INVOKEVIRTUAL FUNZIONE
ISTORE A
```

```
.method FUNZIONE(locA, locB)
                ILOAD locA
                ILOAD locB
                //isub carica in pila il valore
                //della sottrazione
                ISUB
                IFLT L1
                ILOAD locB
        L1: ILOAD locA
                IRETURN
.end-method
```

```
//NUOVO ESERCIZIO

a = 5
b = 7
c = 6
If ( a <= 8 )
        a = b - c
else
        a = b + c


//SVOLGIMENTO

BIPUSH 0x5
ISTORE A

BIPUSH 0x7
ISTORE B

BIPUSH 0x6
ISTORE C

BIPUSH 9
ILOAD A
ISUB
IFLT R1

        ILOAD B
        ILOAD C
        IADD
        ISTORE A
```

```
R1:

        ILOAD B
        ILOAD C
        ISUB
        ISTORE A


// ESERCIZIO NUOVO

a = 2
b = 50

while ( a < b ){
        a = a * 3
        b = * 2
}

BIPUSH 2
ISTORE A

BIPUSH 50
ISTORE B

ILOAD A
ILOAD B
SUB //A - B
IFLT F2 // < 0  // B Più grande
INVOKEVIRTUAL MOLTIPLICAZIONE
ISTORE A
ISTORE B

F2: HALT

.method MOLTIPLICAZIONE (locA, locB)
                BIPUSH 1
                ISTORE I
                ILOAD B
                ILOAD locB
                IFCMPEQ F1
                ILOAD A
                ILOAD locA
                IADD
                ISTORE A
                IRETURN

F1:
```