



MACQUARIE
University

Department of Computing

ITEC625 Fundamentals of Computer Science
Workshop - Control Structures

Learning outcomes

This weeks workshop is about understanding control structures, namely conditions and loops.

1. boolean expressions

Write boolean expressions to check each of the following. You may assume all relevant variables are already declared and initialized to some value.

- (a) if an integer n is in the range 1 to 100 (inclusive on both sides)
- (b) if an integer n is in the range 1 to 100 (inclusive on right side only)
- (c) if an integer n is in the range 1 to 100 (inclusive on left side only)
- (d) if an integer n is in the range 1 to 100 (exclusive on both sides)
- (e) if an integer n is divisible by integer a but not by integer b
- (f) if integer n is either an even number that is more than 20 or an odd number that is less than 5.

2. Flowcharts

Draw flowcharts for each of the following codes. In which cases will your program result in a **Variable may not have been initialized** error? Are there any other mistakes in the program?

(a)

```
int a = 13;
if (a % 13 != 0) {
    a = a - a % 13;
}
```

(b)

```
int max;
int a = 4, b = 6, c = 2;
if (a >= b && a >= c) {
    max = a;
}
else if (b >= c) {
    max = b;
}
else {
    max = c;
}
```

(c)

```
boolean sameOddity;
```

```

if(a && b) {
    sameOddity = true;
}
if(!a && !b) {
    sameOddity = true;
}
if(a && !b) {
    sameOddity = false;
}
if(!a && b) {
    sameOddity = false;
}

```

3. **Tracing control structures - 1** Trace the following code and determine the value of result.

```

int a = 114;
int result = 0;
while(a != 0) {
    if(a % 2 == 1) {
        result = result + 1;
    }
    a = a / 2;
}

```

4. **Tracing control structures - 2** Trace the following code and determine the value of result.

```

int x = 2;
int n = 8;
int result = 1;
for(int i=1; i <= n; i++) {
    result = result * x;
}

```

5. **Tracing control structures - 3** Trace the following code and determine the value of result.

```

int a = 23, b = 49;
int result = 0, power = 1;

while (a != 0 || b != 0) {
    int bit1 = a % 2;
    int bit2 = b % 2;
    if (bit1 != bit2) {
        result = result + power;
    }
    power = power * 2;
    a = a / 2;
    b = b / 2;
}

```

6. Writing control structures - 1

Consider there are three variables, `low`, `high`, `interval` such that:

- $\text{low} \leq \text{high}$, and,
- $\text{interval} \geq 0$

Write a piece of code that computes the following value, and stores it in a variable `sum`.

`low + (low + interval) + (low + 2*interval) + ... high`

* note that it might not go all the way to `high`

For example,

`low = 3, high = 15, interval = 4 -> sum = 3+7+11+15 = 36`

`low = 1, high = 8, interval = 5 -> sum = 1+6 = 7`

`low = -18, high = 20, interval = 12 -> sum = -18+-6+6+18 = 0`

7. Writing control structures - 2

Given an integer $n \geq 1$, write a piece of code that adds integers from 1 to n with the additional constraints that -

- a. any multiple of 5 should not be added, unless the number is also a multiple of 10, in which case, we should add twice of that number to the sum.

For example,

`n = 12, sumNot5 = 1+2+3+4+6+7+8+9+2*10+11+12 = 83`

8. (ADVANCED) Writing control structures - 3

In the lecture we discussed how to determine if a given integer is prime or not. The snippet is provided below:

```
boolean isPrime = true;
for(int i=2; i*i<=n && isPrime; i++) {
    if(n%i==0) {
        isPrime = false;
    }
}
```

The first few primes are:

prime 0: 2

prime 1: 3

prime 2: 5

prime 3: 7

prime 4: 11

prime 5: 13

(In C-base languages - including Java - we number everything starting at zero).

Given an integer $k \geq 0$, write a piece of code that computes prime k .

For example,

`k = 5 -> 13`

`k = 10 -> 31`

`k = 45 -> 199`

`k = 1000 -> 7927`