Keeping Your CA212 Exercises

- Keep a special folder for CA212 exercises
- Grades will be affected by your exercises
- Tutors will award grades for up to date work on a week to week basis
- This forms a part of your continuous assessment.

W2 cont

Use Repetition Structures to Design and Implement Top-Down Stepwise Refinement of an Algorithm

Iteration

- while and counter controlled repetition.
- while and sentinel controlled repetition.
- for repetition

while Repetition Structure

- Programmer may specify that some lines of code are to be repeated while some condition expression is true.

while Syntax

C++ Syntax: while ( <condition> ) <statement>

- Use the while keyword to implement a while loop.
- <statement> executes repeatedly until the value of <condition> is zero.
- The test takes place before <statement> executes. Thus, if <condition> evaluates to zero on the first pass, the loop does not execute.

while Example

```
int product = 2;
while (product <=1000)
{
    product = 2*product;
}
```
**while Syntax Diagram**

```
while (bool expression) compound statement
```

**Programming Errors**

- Writing a condition which never evaluates to false will result in an infinite loop
- C and C++ are case-sensitive, do not write `While` where you intend to write `while`
- Do not put a `;` after the condition, this will result in an infinite loop

**Counter-Controlled Repetition**

- Looking at several variations of a problem of finding the class-average of the programming exam grades

```c
#include <iostream.h>
int main() {
    int total, // sum of grades
         gradeCounter, // number of grades entered
         grade, // one grade
         average; // average of grades
    total = 0;                           // clear total
    gradeCounter = 1;                    // prepare to loop
    while (gradeCounter <= 10) {         // loop 10 times
        cout << "Enter grade: ";        // prompt for input
        cin >> grade;                  // input grade
        total = total + grade;         // add grade to total
        gradeCounter = gradeCounter + 1; // increment counter
    }
    average = total / 10;                // integer division
    cout << "Class average is " << average << endl;
    return 0;   // indicate program ended successfully
}
```

**Programming Errors**

- The counter-control variable will be off by 1 after the loop has completed, if you use it, you may get a bug!
**Sentinel-Controlled Repetition**

- We may not know in advance the size of the class, so the value of `gradeCounter` cannot be set in the code.
- We wish to process an arbitrary number of grades each time the program is run.

**Stepwise Refinement**

1. **Initialise variables**
2. Input the grades, sum them and count the grades
3. Calculate and print class average

```c++
#include <iostream.h>
#include <iomanip.h>

int main() {
    int total, // sum of grades
        gradeCounter, // number of grades entered
        grade; // one grade
    float average; // number for average
    total = 0;
    gradeCounter = 0;
    cout << "Enter grade, -1 to end: " ;
    cin >> grade;  
    while ( grade != -1 ) {
        total = total + grade;
        gradeCounter = gradeCounter + 1;
        cout << "Enter grade, -1 to end: ";
        cin >> grade;  }
    if ( gradeCounter != 0 ) {
        average = (float)total / gradeCounter; // note the cast here ! see type conversion slide
        cout << "Class average is " << setprecision( 2 )<< setiosflags( ios::fixed | ios::showpoint )<< average <<  endl;
    } else
        cout << "No grades were entered" <<  endl;
}
```

**Programming Errors**

- Do not choose a sentinel value which may be a legitimate input value for use inside the loop, we will not get to use it, the loop will exit.
- Floating point arithmetic is done only approximately on computers – `1/3` cannot be represented precisely.
- Do not test floating point numbers for equality, rather test the precision of their differences.

**Nested Control Structures**

The School offers the CA101 programming course. You have been asked to summarise the results of the examinations. A `1` records a pass, a `2` records a fail. You are required to do the following:

1. Input each result after a prompt.
2. Count the passes and fails.
3. Display in summary, number of passes and fails.
4. If more than 8 pass, print message ‘Raise tuition’.

The School offers the CA101 programming course. You have been asked to summarise the results of the examinations. A `1` records a pass, a `2` records a fail. You are required to do the following:

1. Input each result after a prompt.
2. Count the passes and fails.
3. Display in summary, number of passes and fails.
4. If more than 8 pass, print message ‘Raise tuition’.

Our Observations on the Problem

• 10 test results each time, so counter-controlled loop is appropriate
• We can process only a 1 or a 2, if it is not a 1 we will assume it is a 2
• Two counters used, one for passes, one for fails
• After results are processed, must check for more than 8 passes

Stepwise Refinement of Pseudocode

Initialise variables
input 10 grades and count passes and fails
Print summary and decide if fees should be raised

Initialise passes to 0
Initialise fails to 0
Initialise student counter to 1

while student counter less than or equal to 10
input next exam result
if student passed
   add one to passes
else
   add one to fails
add one to student counter
print number of passes
print number of fails
if more than 8 passes
   print 'raise tuition'

#include <iostream.h>
int main()
{
int passes = 0, failures = 0, studentCounter = 1;

while ( studentCounter <= 10 )
{
    cout << "Enter result (1=pass,2=fail): ";
    cin >> result;
    if ( result == 1 )
        passes = passes + 1;
    else
        failures = failures + 1;
    studentCounter = studentCounter + 1;
}

    cout << "Passed " << passes << endl;
    cout << "Failed " << failures << endl;
    if ( passes > 8 )
        cout << "Raise tuition " << endl;
}

do while Repetition Structure

The do statement implements a do ... while loop.
<statement> is executed repeatedly as long as the value of <condition> remains non-zero.
Since the condition is tested after each time the loop executes the <statement>, the loop will execute at least once.
**do while Example**

```
int product = 2;
do 
  product = 2*product;
while (product <= 1000);
```

**Using do while**

- This loop will cause the loop body to be executed at least once.
- When programming advisable to use braces {} on loop body in order to eliminate confusion with while loop construct.
- The conditional or boolean expression must evaluate false at some point in order for the loop to terminate.

**Example do while program**

```cpp
#include <iostream.h>

int main()
{
  int counter = 1;
do 
    cout << counter << " 
  
while (++counter <= 10);
cout << endl;
return 0;
}
```

**for Repetition Structure**

- Handles counter controlled repetition within the language.
- Initialisation, test and increment all handled in one place.

**for Syntax Diagram**

```
for (loop init; loop condition; loop control, var update) 
compound statement
```
Variable decl with init

variable declaration = expression

Loop bool expr

bool expression

Loop control var update

expression

for Example

// Counter-controlled repetition with the for structure
#include <iostream.h>
main()
{
    // Initialization, repetition condition, and incrementing
    // are all included in the for structure header:
    for (int counter = 1; counter <= 10; counter++)
    {
        cout << counter << endl;
        return 0;
    }

Flowcharting for

counter = 1;

counter ++;
counter <= 10

cout << counter << endl;

True

False

// Calculating compound interest
#include <iostream.h>
#include <iomanip.h>
#include <cmath.h>
main()
{
    double amount, principal = 1000.0, rate = .05;
    cout << "Year" << setw(21) << "Amount on deposit" << endl;
    for (int year = 1; year <= 10; year++)
    {
        amount = principal * pow(1.0 + rate, year);
        cout << setw(4) << year
             << setiosflags(ios::fixed | ios::showpoint)
             << setw(21) << setprecision(2) << amount << endl;
    }
    return 0;
}

Compare while and do while

#include <iostream.h>
main()
{
    int counter, grade, total, average;
    total = 0;
    counter = 1;
    while (counter <= 10)
    {
        cout << "Enter grade: 
        cin >> grade;
        total = total + grade;
        counter = counter + 1;
    }
    average = total / 10; // integer division
    cout << "Class average is " << average << endl;
    return 0; // indicate program ended successfully
}
```cpp
#include <iostream.h>

main()
{
  float average;  /* new data type */
  int counter, grade, total;
  total = 0;
  counter = 0;
  cout << "Enter grade, -1 to end ":
  cin >> grade;
  while (grade != -1) {
    total = total + grade;
    counter = counter + 1;
    cout << grade;
    cout << "Enter grade, -1 to end ":
    cin >> grade;
  }
  if (counter != 0) {
    average = (float) total / counter;
    cout << "Class average is %.2f", average;
  }
  else
    cout << "No grades were entered."
  return 0;  /* indicate program ended successfully */
}
```

**Workshop/Tutorial Exercises**

- **Do exercise 2.16**
  - Hint: uses a sentinel controlled loop
- **Do exercise 2.20**
  - Hint: combines
    - iteration (while, do while, for)
    - selection (if)