# Swinburne University of Technology Sarawak

## COS10009 Introduction to Programming – Semester 1 / 2018

## Pointer (Lab 07)

### Core Task 1

#### To Do

**Accessing and de-referencing pointers.**
Download **pointy.c** from BlackBoard. Open the pointy.c program file with Quincy. It contains almost complete code and comments.
- The program declares a *int* variable *num*, *a int* pointer *numPtr* and an *int* array.
- *numPtr* is then assigned the address of *num.*
- The program then prints the value with printf the first time by using *num as a parameter, the second time by using numPtr as a parameter.*
- The value printed must, of course, be the same both times.
- Print the address of the first element in the array
- Print the content of the array name, the address in array name should be the same as the address of the first element in the array.
- Assign the address in array name into *numPtr*
- Print the whole array using *numPtr*

**You must complete the program segments indicated by dots (. . .)**

### Core Task 2

#### To Do

**Pointer (pass by reference)**
In the main() function, declare a int type one dimensional array (the array can be of any size).
Next, construct two functions with the following prototypes:

```
void input(int ar[], int npts);
double calculate(int ar[], int npts, int *gtr);
```

Function input() will prompt user to input integers. The local variable *npts* contains the size of array ar[].
Function calculate() will calculate and return the average (double type) of all the integers in the one-dimensional integer array. Besides, the function will also count the number of integers in the array that are greater than the average. Pointer *gtr* points to a variable in main function. All printf() statements must be written in the main function)
Note: parameter npts in function calculate() is passed by value, and parameter *gtr is pass by reference.

## Core Task 3

### To Do

**Function (Arguments, parameters and pointers)**
Download **RPM.c**
**Background:** The rotational speed of an electric motor is monitored under load. A sensor records the speed in revolutions-per-minute (RPM) every 2 seconds into an integer array.

Write a function that is passed the integer array containing the speeds and passes the maximum, the minimum, and the average RPM values back to the calling program.
Open the startup file RPM.c with Quincy.
The startup file contains comments giving the decomposition outline of the program and RPMstats.
The array containing the RPM data is hard-coded at the start of main for simplicity, and the size of the array is declared in a pre-processor #define.
This time you must complete most of the program on your own. Ask the tutor at anytime if unsure.
- Complete, compile, link **RPM.c**, then **test.**
The average RPM should be 1952.4 RPM.

## Vital Task

### To Do

Complete the following program according to the comments provided:

```
/* This program prints the summary of BMI for a group of students. */

#include <stdio.h>
#define SIZE 10

/* Write a function
     void bk_input(double b[SIZE]) { ... }
     where b[] is the array, use for loop to get input from students on their
     weight(kg) and height(meters), then calculate the BMI (BMI = W / H^2) for each
     student and store it into array b[]*/

/* Write a function
     void disp_bk(double b[SIZE]) { ... }
       where b[] is the array
       Print the entire array to the screen. */

/* Write a function
       void summary(double b[SIZE], int *underw, int *ideal, int *overw, int *obese) {
       ... }
       where b[]is the array, *underw, *ideal, *overw and *obese are pointers, use a
       for loop to count the number of students fall under each category base on the
       following criteria: BMI<20 is underweight, BMI between 20 and 25 is ideal, BMI
       above 25 but less than 30 is considered overweight and BMI over 30 is considered
       obeses*/

/* Write a function
       void dispsummary(double b[SIZE], int uw, int idl, int ow, int obs) { ... }
       where b[]is the array, underw, idl, overw and obese are variables. Print the
       following:
       printf("Number of students underweight:%i\n", . . .);
       printf("Number of students with ideal BMI:%i\n", . . .);
       printf("Number of students overweight:%i\n", . . .);
       printf("Number of students obese:%i\n", . . .);


int main(void)
{
    /* declare needed variables or constants, e.g. */
       int no=0, uweight=0, ideal=0, oweight =0, obese=0;
     double average;

       double bk[SIZE];

     /* call bk_input(...) function */

     /* call the disp_bk(...) function */

     /* call the summary(...) function */

     /* call the dispsummary(...) function */

     /* Return to the operating system */
`
```