Use the Visual Studio Debugger to help answer the three questions below. In each case write an explanation of what is happening in each question. Upload your explanations using the link provided in Moodle.

Q1: The following code contains three serious memory bugs, when run it usually loops continuously until it either ends in a memory access violation fault or by using up all the available memory on the system. Identify, solve and explain each of the three bugs. If the programme still has run time errors you have not found all the bugs.

// this program is buggy

#include <iostream>

using namespace std;

void main()

{

double\* d = new double;

for(unsigned int i = 0; i < 3; i++) {

d[i] = 1.5 + i;

}

for(unsigned int i = 2; i >= 0; i--) {

cout << d[i] << endl;

}

}

* Bug 1: Incorrect syntax when declaring pointer. Correct syntax: double \*d;
* Bug 2: No place in memory was allocated for the pointer. We can fix this by declaring a size for the pointer: double \*d = new double(3);
* Bug 3: The program triggers a breakpoint as there is no System(“pause”) declared:

for (int i = 0; i < 3; i++) {

d[i] = 1.5 + i;

}

for (int i = 2; i >= 0; i--) {

cout << d[i] << endl;

}

system("pause");

Q2: What does this program output and why?

void main()

{

char chr = 'N';

if (chr == 'Y' || 'y')

cout << "chr is y" << endl;

else if (chr == 'N' || 'n')

cout << "chr is n" << endl;

else

cout << "chr is something else" << endl;

}

This program output chr is y because the variable chr has not been initialized correctly as a pointer and will therefore not store the value ‘N’. If memory was allocated to the variable chr, then it could take the value ‘N’;

Q3: The following program compiles and runs. It outputs 5 and 3. Explain in detailed steps what this program is doing.

#include <iostream>

using namespace std;

void main()

{

int t1[] = {0,0,1,1,1}, t2[] = {0,0,1,1,1};

int \*p1 = t1, \*p2 = t2;

while (!\*p1++ || !\*p2++);

cout << (p1-t1) << endl;

cout << (p2-t2) << endl;

}

* Two integer arrays are created (t1[] and t2[]).
* Values are assigned to both arrays.
* Two integer pointers are created (\*p1 and \*p2).
* Both pointers are initialized with the size of the arrays.
* A while loop is then executed while (!\*p1++ || !\*p2++).
* The program then prints out (p1 – t1) and (p2 – t2)
* These values = 5 and 3 because p1 now evaluates to 5 as all 5 spaces in the array have been initialized to p1.
* While p2 = 3 as only the spaces with a 1 have been initialized to p2.
* t1 and t2 = 0, because they do not hold values because they are arrays.

Hint: Read Kernighan & Ritchie, The C Programming Language, 2nd Edition, Prentice Hall PTR, 1988, p. 166, Section A.7.7 Additive Operators, last paragraph.

Hint 2: Read Kernighan & Ritchie, The C Programming Language, 2nd Edition, Prentice Hall PTR, 1988, p. 48, Section 2.12 Precedence and Order of Evaluation (Table 2.1)

Q4: Are the following two programs equivalent? Explain the behaviour and output from these two programs. Both programs compile and run.

Program (1)

#include <iostream>

using namespace std;

int main()

{

int count = 1;

for (; count <= 5 ; count++)

{

int count = 1;

cout << count << "\n";

}

return 0;

}

Program (2):

#include <iostream>

using namespace std;

int main()

{

int count = 1;

while (count <= 5)

{

int count = 1;

cout << count << "\n";

count++;

}

return 0;

}

* Program 1 runs through the for loop 5 times and prints out the value 1 through each iteration.
* Program 2 is an infinite loop as the variable count is initialized twice. It is initialized before the while loop and inside the while loop. This means that count will constantly have a value of 1 or 2 as it is incremented at the end of the loop. The loop is infinite as count will always be less than 5.