CS2002 Tutorial Week 4

C programming language

1. Write down a line of C-code that calls the void function below with the appropriate argument to alter the value of a local variable int n; of the block containing the call. Describe the outcome of the function call.

```
void func(int *ip) {
  *ip *= 2;
}
```

2. What is "undefined behaviour". Why should it be avoided? Give two examples of code which lead to undefined behaviour. What is "implementation defined behaviour"? Give an example.

3. In a computer using two's complement in its CPU, there is no difference between the hardware operations of addition of signed and unsigned integers.

   In C, an overflow on integers is undefined behaviour. (i) What are potential consequences? (ii) [Open-ended] What strategies might you devise to resolve this undefined behaviour? **Hint:** The C standard library provides `<limits.h>` that always contains INT_MAX and INT_MIN, the largest and smallest possible values that can be stored in an int.

4. Pointer notation and its arithmetic may be used to access arrays and the array notation can be used to access storage referenced by pointers. Given the definitions `int ia[100]; int *ip = &ia[0];` explain which of the following expressions are valid and which are not:

   (a) `*(ia+5)`

   (b) `ip[5]`

   (c) `ip = ia + 0`

   (d) `ip = ia`

   (e) `ia = ip`

   (f) `++ia`

   (g) `++ip`

5. Why is the following a valid C program?

```
int main(void) {
   http://www.google.com
   return 0;
}
```

6. A C programmer is told to write a program that reads a number, and then print it back out. That person writes the following code. The programmer says that when the code is compiled it works correctly, so they deserve full marks. What do you think?

```
#include <stdio.h>

int* get_ptr() {
  int i;
  return &i;
}

int main(void) {
  printf("Enter a number:");
  scanf("%d", get_ptr());
  printf("Pointer value is: %d\n", *(get_ptr()));
}
```

7. What is the difference between a struct and a union? Give an example of somewhere you would use a struct, and somewhere you would use a union. Why is it a good idea to have a function which sets up the initial values of structs?

   Explain the meaning of the following code, and how you could use both `S` and `intval` after this line. Why is this a useful and common thing to write?

   ```
   typedef struct S {int i,j; } Intval;
   ```