

R249/102

DUBLIN INSTITUTE OF TECHNOLOGY
KEVIN STREET, DUBLIN 8.

BSc Information Systems / Information Technology

Stage 1

SUPPLEMENTAL EXAMINATIONS 2007

PROGRAMMING AND ALGORITHMS

MR. C. O'LEARY
DR. B. O'SHEA

SEPTEMBER 2007, 3 HOURS

ATTEMPT **4** QUESTIONS

ALL QUESTIONS CARRY EQUAL MARKS

1. (a) Consider the code shown below:

```
#include <stdio.h>

int main() {

    FILE *file_pointer;
    file_pointer = fopen("a_file", "a");
    person p1 = { "Bertie Ahern", 55};
    person p2 = { "Tony Blair", 53};
    person p3 = { "George Bush", 60};

}
```

This program is supposed to create 4 person structure instances and write them to the file named `a_file`.

Show how to change the code so that the desired behaviour is produced.

(6 marks)

- (b) With reference to file handling and the example from part (a) above, provide the necessary code to do the following:

- i. Read the three records from the file and print them out.
- ii. Randomly access the second record, read it and print it out.
- iii. Overwrite the second record with a new record.

(9 marks)

- (c) Devise and clearly describe a method for maintaining a linked list in persistent storage (e.g. a file). Consider all situations including addition of a new element at the start, in the middle and at the end of the list, as well as deletion of an element from the list.

(10 marks)

2. (a) Consider the code shown below:

```
#include <stdio.h>

int main() {

    char a;
    short b;
    int c;
    unsigned int d;
    double e;

    a = 1;
    b = 2;
    c = 3;
    d = 4;
    e = 5;

    printf("a: %d\n", sizeof(a));
    printf("b: %d\n", sizeof(b));
    printf("c: %d\n", sizeof(c));
    printf("d: %d\n", sizeof(d));
    printf("e: %d\n", sizeof(e));

    printf("a: %d\n", a);
    printf("b: %d\n", b);
    printf("c: %d\n", c);
    printf("d: %d\n", d);
    printf("e: %d\n", e);

}
```

Will this program compile? If so, what output would you expect when it runs?

Explain clearly.

(6 marks)

- (b) With reference to the C programming language, answer the following 9 questions (1 mark each). Use sample code where possible to make your answer clear.

- i. What is a *replacement character*?
- ii. Describe how the `scanf(...)` function is used?
- iii. Describe how the `gets(...)` function is used?
- iv. Why is a typecast usually necessary after a call to `getchar()`?
- v. How is the `+=` operator used?
- vi. How is the `++` operator used?
- vii. What will happen if the `++` operator is used on a char?
- viii. What words cannot be used as variable names?
- ix. What is the maximum length of a variable name?

(9 marks)

- (c) Present a detailed discussion of the open-source movement.

(10 marks)

3. (a) Consider the incomplete code shown below:

```

#include <stdio.h>

int main() {

    int i;
    int limit;
    limit = argv[1];

    while(i = 0; i <= limit; i++) {
        if(i / 3 = 0) {
            printf("%d is divisible by 3\n", i);
        }
    }
}

```

This program is supposed to take a number (the *limit*) as a command line argument, and print out all the numbers divisible by 3 between 0 and this limit – as shown below:

```

c:\> q2a 20
0 is divisible by 3
3 is divisible by 3
6 is divisible by 3
9 is divisible by 3
12 is divisible by 3
15 is divisible by 3
18 is divisible by 3

```

The program shown contains a number of errors. Show how to change the code so that the desired output is produced.

(6 marks)

- (b) Consider the scenarios below where the user is asked to enter values until some condition is reached.

You are required to select the most appropriate type of loop, and provide the C code for the loop (you can use pseudocode for reading, writing etc). You must also briefly justify your choice of loop type.

- i. The user must select a menu item between 1 and 10. If, and only if, they enter an incorrect value, the loop should iterate another time.
- ii. The user must enter exactly 10 values.
- iii. The user must enter values until they elect to end the loop by entering a sentinel value.

(9 marks)

- (c) Describe in detail the operation and behaviour of *stack* and *queue* data structures. Show how *one of* these data structures could be implemented using an array, using either *pseudocode* or *C code*.

(10 marks)

4. (a) Consider the code shown below:

```

#include <stdio.h>

int main() {

    int person1height;
    int person2height;
    int person3height;

    printf("Enter height for person 1: ");
    scanf("%d", &person1height);
    printf("Enter height for person 2: ");
    scanf("%d", &person2height);
    printf("Enter height for person 3: ");
    scanf("%d", &person3height);

    printf("Person 1 height: %d\n", person1height);
    printf("Person 2 height: %d\n", person2height);
    printf("Person 3 height: %d\n", person3height);
}

```

This program prompts the user to enter the heights of 3 people, and then stores and prints out these heights.

Rewrite the code so that it can input, store and print out the heights of any number of people, where the number of people is defined in a preprocessor command, **(6 marks)**

(b) With reference to the C programming language, answer the following 9 questions (1 mark each). Use sample code where possible to make your answer clear.

- i. How is a 2-D array declared?
- ii. What is the maximum number of dimensions of an array?
- iii. How are arrays declared inline?
- iv. How is an array declared using pointer notation?
- v. What is the behaviour of the `malloc` function?
- vi. What does the replacement character `%-20.6s` produce in a `printf` statement?
- vii. What is the behaviour of the `strcmp` function?
- viii. How is the end of a string stored in memory?
- ix. How does the `strcat` function concatenate strings?

(9 marks)

(c) Provide pseudocode for the quicksort algorithm, and use it to sort the following numbers:

46 21 58 47 87 68 21 52

(10 marks)

5. (a) Consider the code shown below:

```
#include <stdio.h>

int main() {
    unsigned int select(unsigned int, unsigned int);
    printf("Lotto combinations: %u\n", select(42, 6));
}

unsigned int select(unsigned int x, unsigned int y) {
    unsigned int combine(unsigned int, unsigned int);
    unsigned int factorial(unsigned int);
    return combine(x, x - y) / factorial(y);
}

unsigned int combine(unsigned int number, unsigned int floor) {
    if(number == floor) return 1;
    return number * combine(number - 1, floor);
}

unsigned int factorial(unsigned int number) {
    if(number == 1) return 1;
    return number * factorial(number - 1);
}
```

This code calculates the total amount of combinations possible in a lottery where 6 numbers must be selected from 42 (the total number of combinations, incidentally, is 5,245,786).

In relation to the above code and C programming in general, explain the following:

- i. The meaning of *recursion*
- ii. Why recursion is used
- iii. The dangers of using recursion
- iv. The motivation for the use of `unsigned int` data types
- v. The requirement for function prototypes
- vi. The multiple return statements in the code above

(6 marks)

- (b) Rewrite the code from part (a) so that it calculates the same result without recursion.

(9 marks)

- (c) Explain what is meant by pseudo-random numbers, and provide the code necessary to generate simple pseudo-random numbers using the Fibonacci series.

(10 marks)

6. (a) Consider the code shown below:

```
#include <stdio.h>

int main() {

    char nationality;

    printf("Enter your nationality\n");
    printf(" (A)merican\n");
    printf(" (B)ritish\n");
    printf(" (C)anadian\n");
    printf(" (D)anish\n");
    printf("\t--> ");
    nationality = (char)getchar();

    if(nationality == 'A' || nationality == 'C')
        printf("You cannot vote in our elections\n");
    else if(nationality == 'D')
        printf("You can vote in our European elections\n");
    else if(nationality == 'B')
        printf("You can vote in our European
                and Local elections\n");
    else
        printf("Invalid choice\n");
}
```

This program allows non-Irish citizens find out their voting rights in Ireland. Rewrite the program using a `switch` statement instead of `if...else...if` statements and change the program behaviour so that users who enter an invalid choice are given another opportunity to enter their nationality.

(6 marks)

- (b) With reference to the C programming language, answer the following 9 questions (1 mark each). Use sample code where possible to make your answer clear.

- i. What is meant by a *static* variable?
- ii. What is meant by an *auto* variable?
- iii. What is meant by a *register* variable?
- iv. What is meant by a *global* variable?
- v. What is meant by the ASCII character set?
- vi. What is meant by a *file pointer*?
- vii. What is meant by *file permissions*?
- viii. What is meant by a *binary file*?
- ix. What is meant by a *text file*?

(9 marks)

- (c) Discuss the importance of testing in software development projects, and describe three approaches to software testing.

(10 marks)