

UNIVERSITY COLLEGE DUBLIN

NATIONAL UNIVERSITY OF IRELAND, DUBLIN

An Colaiste Ollscoile Baile Atha Cliath

Ollscoil na hEireann, Baile Atha Cliath

AUTUMN EXAMINATIONS 2005

FIRST EXAMINATION IN ENGINEERING

Programme Codes: ENBDF0002, ENBDF0003,
ENBDF0004, ENBDF0005, ENBDF0008, ENBDF0011

COMP1604 COMPUTER SCIENCE

Prof. J. Kramer

Prof. B. Smyth

Dr. J. Murphy*

Time allowed: 2 hours

Answer **Question 1** and **one** other Question.

Question 1 carries 60 marks; Questions 2 and 3 carry 40 marks.

This is a closed-book examination. No calculators allowed.

Loose Rough Work sheets are not to be distributed or used.

READ EACH QUESTION CAREFULLY.

Question 1 (COMPULSORY) [60 marks]

Answer all parts (a) – (t). Each part carries 3 marks.

- (a) True or False (*no explanation required*): the term “software” refers to **programs** which control the computer and allow the user to perform useful tasks.
- (b) True or False (*no explanation required*): in a C program, comments **must** be included to explain and document what the program does, and why.
- (c) True or False (*no explanation required*): a logic error in a C program will be **detected** by the compiler, although the error message may be difficult to understand.
- (d) If $x=4$, $y=4$, and $z=2$, what is the value of w in the expression $w=x+y/z$?
- (e) What is the screen output of the following fragment of C code (*no explanation required*):

```
float x=2.23607;
printf("value is %.2f\n",x);
```

- (f) What is the screen output of the following fragment of C code (*no explanation required*):

```
int i=-1,j=2;
if (j<=i){
    printf("normal case\n");
} else if ((-i)>(j/2)){
    printf("exceptional case\n");
} else {
    printf("error case\n");
}
```

- (g) What is the screen output of the following fragment of C code (*no explanation required*):

```
int a=0, b=2, c=-1;
if ((!a) && ((b*c)>0) ){
    printf("condition is true\n");
} else {
    printf("condition is false\n");
}
```

- (h) What is the screen output of the following fragment of C code (*no explanation required*):

```
int i;
for (i=20;i>0;i--){
    i = i/3;
    printf("i is %d\n",i);
}
```

[Question 1 continues]

Question 1 (continued)

- (i) What is the screen output of the following fragment of C code (*no explanation required*):

```
int i;
for (i=0;i<=3;i++){
    switch(i){
        case 2: printf("2\n");
                break;
        case 3: printf("3\n");
                break;
        default: printf("default\n");
                break;
    }
}
```

- (j) Select the correct answer: If a function has a return type **int**, this means

- (j-1) it takes a single parameter of type **int**.
- (j-2) it can manipulate integer-valued variables only.
- (j-3) it returns an integer value to the calling function.

- (k) Select the correct answer: The function prototype

```
float fname(int *x);
```

tells us that

- (k-1) **fname()** takes 1 argument of type pointer-to-int and returns a value of type float.
 - (k-2) **fname()** takes 1 argument of type int and returns a value of type float.
 - (k-3) **fname()** takes 1 argument of type float and returns a value of type pointer-to-int.
- (l) What is the screen output of the following C program (*no explanation required*):

```
#include <stdio.h>
int f(int i, int j){
    return (i*j);
}
void main(void){
    printf("result is %d\n", f(-2,f(-2,f(-2,2))) );
}
```

- (m) Select the correct answer: If you want to add the first and second elements of an array **arrints**, you should use

- (m-1) **arrints[1]+arrints[2]**
- (m-2) **arrints[0]+arrints[1]**
- (m-3) **arrints[1+2]**

- (n) True or False (*no explanation required*): In C, all elements of an array **must** have the same type.

[Question 1 continues]

Question 1 (continued)

- (o) What is the screen output of the following fragment of C code (*no explanation required*):

```
int i;
int array[5] = {1,2,3};
for (i=0; i<=4; i++){
    printf("element number %d is %d\n", i+1, array[i]);
}
```

- (p) Select the correct answer: If the pointer `ptr1` currently points to `x` and you want to assign the current value of `x` to `y`, you could use the statement

(p-1) `ptr1 = &y;`
(p-2) `*ptr1 = y;`
(p-3) `y = *ptr1;`

- (q) What is the screen output of the following fragment of C code (*no explanation required*):

```
int x = 1;
int y = -2;
int* p = &x;
*p = (*p)*y + (*p)*x;
printf("x is %d and y is %d\n", x, y);
```

- (r) True or False (*no explanation required*): in C, the **name** of a string is treated by the compiler as a pointer-to-char variable whose value can be reassigned in the program.

- (s) What is the screen output of the following fragment of C code (*no explanation required*):

```
char str[]="abcdefghijklmn";
char vowels[]="aeiou";
int i, j;
for (j=0; vowels[j]!='\0'; j++){
    for (i=0; str[i]!='\0'; i++){
        if (str[i]==vowels[j]){
            str[i]='Y';
            break;
        }
    }
}
printf("string=%s\n", str);
```

- (t) Select the correct answer: `fopen("datafile.txt","r")` means

(t-1) if it exists, open `datafile.txt` for random access, otherwise return an error.

(t-2) if it exists, open `datafile.txt` for reading only, otherwise return an error.

(t-3) if it exists, open `datafile.txt` for reading only, otherwise open the first file found and read from it.

Question 2 [40 marks]

Answer all parts (a) – (c).

(a) The following fragment of C code adds up the integers from 1 to 5 inclusive using a **while** loop:

```
int sum = 0, i = 1;
while (i <= 5)
    sum = sum + i;
    i++;
}
```

Re-write this code fragment using a **for** loop instead of the **while** loop.

(b) The following fragment of C code determines the type of fuel **f** using **if/else-if/else** statements:

```
char f;
/* suppose a value is now entered for f - code not shown */
if (f=='u'){
    printf("unleaded petrol\n");
} else if (f=='p'){
    printf("premium petrol\n");
} else if (f=='d'){
    printf("diesel\n");
} else printf("incorrect value entered\n");
```

Re-write this code fragment using a **switch** statement instead of the **if/else-if/else** statements.

(c) Consider the following C program:

```
#include <stdio.h>
/* DEFINITION OF FUNCTION "zerofinder" GOES HERE */
void main(void) {
    int i, array1[8]={1,-1,-1,0,1,0,-1,1};
    i = zerofinder(array1,8);
    if (i == -1) printf("\nno zero value in array1\n");
    else printf("\nfirst zero element of array1 has index %d\n", i);
}
```

Write down the definition of function **zerofinder()** which returns the index of the first 0 value, or **-1** if no 0 value is found, so that the output of the above program is:

first zero element of array1 has index 3

Question 3 [40 marks]

Answer parts (a) and (b).

- (a) Consider the following C program:

```
#include <stdio.h>
void main(void)
{
    int intarr[4], i, total=0;
    for (i=0 ; i<4; i++){
        printf("enter value number %d: ", i+1);
        scanf("%d", &intarr[i]);          /* LINE 1 */
        total += intarr[i];              /* LINE 2 */
    }
    printf("sum of inputs is %d\n", total);
}
```

Re-write the lines **LINE 1** and **LINE 2** using “array pointers” instead of array subscripts.

- (b) Consider the following C program:

```
#include "stdio.h"
void main(void){
    char message[80]="I love C programming";
    int i=0, count=0;
    while (message[i]!='\0'){
        if ((message[i]=='c') || (message[i]=='C')){
            count++;
        }
        i++;
    }
    printf("\n%s\n" contains %d c's\n", message, count);
}
```

Re-write the above program so that all the code for determining the number of **c**'s in the string **message** is contained in a function called **c_counter()** that you should define, while the declaration and initialization of **message** and the output is still done from **main()**.

o O o