Faculty of Science – School of Physics

PHYS2020 – Computational Physics

Mid-session Test Example

Duration : 50 minutes   Contribution towards final assessment : 16%

Answer all questions. Each question is of equal value. This is a closed book exam.
Use black or blue pen only.

NOTE: the real exam won't have this many questions.

Question 1

With respect to representing floating-point numbers in a series of bytes:

- what is a mantissa?
- what does "NaN" mean?
- explain the tradeoff between the number of digits of precision and the range of the exponent.

Question 2

Suppose you wanted to run a program called "mytest" that was stored in the directory one level above your current working directory, and you wanted the program to read input from the file "in.txt" in your current working directory, and write output to the file "/tmp/out.txt".

Write down the GNU/Linux command-line to do this.

Question 3

Write concise descriptions of the following concepts:

- Metadata.
- Standard error.
- The shell.
- The exit status of a program.

Question 4

What is wrong with the following function to convert degrees C to Farenheight?

```
double toFarenheight(double celcius) {
   return celcius * (9/5) + 32;
}
```

How would you fix it?

Question 5

How many times will "statement" be executed in this loop:

```
for (int i = 1; i > -1; i--) {
    statement;
}
```

How about this loop?

```
for (int i = 0; i > -1; --i) {
    statement;
}
```

And this one?

```
for (int i = 10; i++; i < 10) {
    statement;
}
```

And this one?

```
for (unsigned char i = 0; i < 256; i++) {
    statement;
}
```

Question 6

Using the information about precedence/associativity at the end of this question, evaluate each of the following C expressions, giving your answer as a base 10 number.

```
2 + 3 * 6 + 2;
2 >> 2 * 3;
3 * 2 >> 2;
1 | 4 + 1;
! 1 && ! 0 < 1;
2 / 3 + 1 & 1;
-1 - -1 * -1;
0644 & 0x111;
```

The precedence of an operator gives the order in which operators are applied in expressions: the highest precedence operator is applied first, followed by the next highest, and so on.

The associativity of an operator gives the order in which expressions involving operators of the same precedence are evaluated.

The following table lists all C the operators, in order of precedence, with their associativity:

Operator	Associativity
`() [] ->> .`	left-to-right
`- + ++ -- ! ~ * & sizeof (type)`	right-to-left
`* / %`	left-to-right
`+ -`	left-to-right

`<< >>`	`left-to-right`
`< <= > >=`	`left-to-right`
`== !=`	`left-to-right`
`&`	`left-to-right`
`^`	`left-to-right`
`|`	`left-to-right`
`&&`	`left-to-right`
`||`	`left-to-right`
`?:`	`right-to-left`
`= += -= *= /= %= &= ^= |= <<= >>=`	`right-to-left`
`,`	`left-to-right`

Note: the + - & and * operators appear twice in the above table. The unary forms (on the second line) have higher precedence that the binary forms.

Operators on the same line have the same precedence, and are evaluated in the order given by their associativity.

Question 7

Assuming the following declarations:

```
int  *pi;
int   i,j;
char *pc;
char  c[]="aardvark";
```

what are the final values of i, j, and c after executing the following statements?

```
i   = 1;
j   = 2;
pi  = &i;
*pi = 3;
pc  = c;
pc += 3;
*pc = 'R';
--j;
```

Question 8

The following code fragment shows a potentially infinite loop.

```
int i = 0;

while (1) {
    i++;
    statement;
}
```

Replace "statement" with a C statement that will cause the loop to be executed precisely 100 times.

Question 9

Compare and contrast “static” and “automatic” variables in C. Give simple examples of each.

Question 10

In the following program fragment, give a concise description of what is happening at the line beginning with “assert”.

```
#include <stdio.h>
#include <assert.h>
FILE *fp;

assert(NULL != (fp = fopen("data.dat", "r")));
```

Question 11 - difficult

Write a C function that returns the number of odd numbers between its two integer (C “int” data type) arguments. Examples: your function should return the following results:

```
   arg1    arg2     function result
    1      -2           1
    1       5           1
    2       4           1
    4       2           1
    0      10           5
   -1       0           0
    1       1           0
```

Question 12

Given the following “struct” definition for a time during the day:

```
struct time {
    int hour;
    int minute;
    float second;
}
```

Write a function that begins like this

```
void addTimesAndPrint(struct time t1, struct time t2)
```

and which prints the result of adding the time t1 to the time t2. You should use “assert” to ensure that t1 and t2 are both valid times within a day (i.e., that that are between 00:00:00.000 (inclusive) and 23:59:60.00 (exclusive)). The result time may overflow one day (i.e., the hour may go from 0 to 47).

Examples are:

```
         t1               t2          function result

    {1,  30, 10.0}  {2,  30, 51.1}      04:01:01.1
    {23, 59, 59.9}  {23, 59, 59.9}      47:59:59.8
    {1,  30, 10.0}  {-1, 30, 61.0}   assertion failure
```

Question 13 - difficult

The C library function

```
char *strcat(char *dest, char *src)
```

appends the src string to the dest string, overwriting the null byte at the end of dest, and then adds a terminating null byte. You may assume that the strings do not overlap. The function returns a pointer to the dest string. The dest string must have sufficient space allocated to hold the resulting string.

For example,

```
char dest[100]="one";
char src[]="two";

strcat(dest,src);
```

will result in dest containing “onetwo”.

Write an implementation of strcat (i.e., write a function that could be used to replace the C library function).