ESc101N: Fundamentals of computing(First mid-semester Exam)

August 28, 2009

Section:
Name:
Roll Number:

Question:	1	2	3	4	5	6	Total
Points:	5	5	5	5	5	5	30
Score:							

Instructions

1. Read these instructions carefully.
2. Write you name, section and roll number on all the pages of the answer book.
3. Write the answers cleanly in the space provided. You can ask for extra sheets for rough work.
4. Using pens (blue/black ink) and not pencils. Do not use red pens for answering.

Question 1. Write your answer on the line provided.

(a) (1 point) What is the value of the variable `x` in after the following assignment.

```
int x = 10 - 5 - 7 / 4 * 4;
```

> **Solution:** The value is `1` because `(10 - 5) - (7/4) * 4 = 1` Marking scheme: No partial marking.

(b) (2 points) Bracket the `C` expression `x < 10 || - z * y - 4` so as to show in which order the compiler evaluates the expression.

> **Solution:**
>
> `(x < 10) || ((- z) * y) - 4)`
>
> Marking scheme: 0.5 mark for each correct bracket.

(c) (2 points) From the `C` expression `(- 2) * (x - 3) < x + (2 * z) && (z < 10)`, remove all the unnecessary parenthesis so that the meaning does not change.

> **Solution:** `- 2 * (x - 3) < x + 2 * z && z < 10` Marking scheme: 0.5 marks each for removing/preserving each bracket.

Question 2. (5 points) What is the output of the following program?

```
#include<stdio.h>
int x=0;
int main()
{
    printf("%d\n",x);
    int x = 5;
    printf("%d\n",x);
    {
      int x = 10;
      printf("%d\n", x);
      x++;
    }
    printf("%d\n",x);
    x++;
    printf("%d\n",x);
    return 0;
}
```

> **Solution:**
>
> ```
> 0
> 5
> 10
> 5
> 6
> ```

Question 3. (a) (2 points) Give the most appropriate declaration for the `C` function "`foo`" so that the program compiles without errors.

```
int main()
{
  double x;
  x = foo(10, 'a');
}
```

Solution:

```
double foo(int, char);
```

However it could also be `int foo(int, int);` because of the automatic conversions in `C`. Since we have asked for the most appropriate declaration for `foo` give 2 marks for the former and 1 mark for the latter (or something similar). Don't cut marks for missing semicolon but other wise cut marks for any syntax error.

(b) (3 points) The contents of the file `main.c` and `foo.c` is given below.

```
/* main.c */
int foo(int x, int y);
int bar(int);
int main()
{
  foo(2,3);
  bar(3);
  return 1;
}
```

```
/* foo.c */

int foo(int u, int v)
{
  return 42;
}
```

The following commands are used to compile the above files.

```
$ gcc -c main.c
$ gcc -c foo.c
$ gcc main.c foo.c
```

Which all, if any, of these compilations will fail and what kind of errors, if any, will be reported by the compiler.

Solution: Only the last compilation leads to error. The error is a linker error as the function bar is not defined any where.

Marking scheme: 2 marks if they say that the last statement is the only one that will create a problem. 1 additional mark if they say that it will be a linker error or link time error or any equivalent thing. No partial marks if they say that steps 1 and 3 will give error. The student is expected to know that step 1 and 2 does not give error. (difference between compilation only and linking).

Question 4. (5 points) What will be the output of the following program ?

```
#include<stdio.h>
int x = 100;
void bar(int);

int main()
{
  printf("%d\n",x);
  bar(3);
  printf("%d\n",x);
}

void bar(int x)
{
  if( !x ) return;
  printf("%d\n",x);
  x--;
  bar(x);
}
```

Solution:

```
100
3
2
1
100
```

Question 5. (5 points) Recall the Taylor series expansion for $\log(1+x)$.

$$\log(1+x) = x - \frac{x^2}{2} + \frac{x^3}{3} - \ldots.$$

Given below is the `C` code that to compute an approximation $\log(1+x)$ to n terms (the first term is x and so on). It takes two parameters x and n where n is the number of terms in the summation and returns the result.

```
double log(double, int);
double log(?1?, ?2?)
{
  double sum, term; \\
  term = ?3?;
  sum  = 0;
  for( int i = ?4?  ; i <= n ; i ++)
  {
    sum += term / i;
    term = ?5?;
  }
  return sum;
}
```

Fill in the blanks with appropriate values.

Solution:

```
double log(double x, int n)
{
  double sum, term;
  term = x;
  sum  = 0;
  for( int i = 1  ; i <= n; i ++)
  {
    sum = sum + term / i;
    term = - term * x;
  }
  return sum;
}
```

Question 6. (5 points) Let $\{a_n\}_{i=0}^{\infty}$ be a series of real numbers. We say that a_n satisfies a linear recurrence of degree 2 if there are reals α and β such that

$$a_{n+2} + \alpha a_{n+1} + \beta a_n = 0$$

Given the initial two values a_0 and a_1 the series is completely determined. Fill in the details of the following C function which takes as parameters the coefficients α and β, and the initial values a_0, a_1 and computes the value of a_n for all n.

```
double linear(?1?);
double linear(double alpha, double beta, double a0, double a1, int n)
{
  /* solving eqn a_{n+2} + \alpha a_{n+1} + \beta a_{n} = 0 */
  double a2;
  while( ?2? )
  {
    a2  = ?3?;
    a0 = ?4?;
    a1 = ?5?;
    n --;
  }
  return a0;
}
```

Solution:

1. `double linear(double, double, double, double,int);`. Any answer to the same spirit is acceptable.
2. `n > 0`
3. `a2 = - beta * a0  - alpha * a1;`
4. `a0 = a1;`
5. `a1 = a2;`

The complete solution is

```
double linear(double, double, double, double, int);
double linear(double alpha, double beta, double a0, double a1, int n)
{
  double a2;
  while( n > 0 )
  {
    a2 = - beta * a0  - alpha * a1;
    a0 = a1;
    a1 = a2;
    n --;
  }
  return a0;
}
```