ICS'12 Homework 4

1. Translate the following assembly into C codes. Note that you can name local variables represented by 28(%esp), 24(%esp)… freely as you like.

```
    movl    $1, 28(%esp)
    movl    $1, 24(%esp)
    movl    $10, 16(%esp)
    movl    $3, 20(%esp)
    jmp L2
L3:
    movl    28(%esp), %eax
    movl    %eax, 12(%esp)
    movl    24(%esp), %eax
    movl    %eax, 28(%esp)
    movl    12(%esp), %eax
    addl    %eax, 24(%esp)
    incl    20(%esp)
L2:
    cmpl    $10, 20(%esp)
    jle   L3
    (Following code omitted…)
```

2. Translate the following switch statements into assembly **using jump table**.

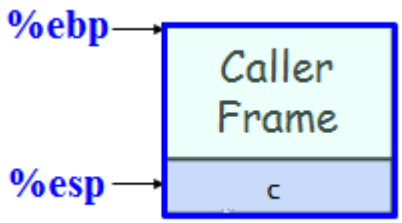
```
int x = <some value>;
int result = 0;
switch (x) {
    case 64:
        result = x + x;
        break;
    case 66:
        result = x + 10;
        break;
    case 68:
        result = x * 2;
    // Notice: there is no break here!
    case 69:
        result = result + 1;
        break;
```

```
    default:
        result = 3;
        break;
}
```

3. Describe why and how stack frames are chained. You could present a graph for illustration.

4. Draw a stack and indicate where %ebp and %esp points **for each step** when the following function is called. Here is an example where the first step takes place.

int foo(int a, int b, int c);



Programming Exercises

Here are another two C programming exercises which you should also finish this week. And you should submit your answers to SVN server *svn://ipads.se.sjtu.edu.cn/ics-se11/ics<student no.>/tut1*.

The filename for the two problems should be ex4-1.c and ex4-2.c respectively. **Do not write them on your exercise book!**

1. Implement a function *reverse(s)* that gives a reverse string of the

original string. You should give your own test example in *main()*. **Do not modify the original string.**

```
/*
  * example: reverse(“abc”) should return “cba”.
  */
char *reverse(const char *s);
```

2. Comple the following C codes that generates a 10x10 matrix (a_{ij}), where $a_{ij} = i+j$.

Hint:

(1) Use *malloc()* or *calloc()* for space allocation.

(2) ‘*int a[M][N]; p = a*’ or ‘*int a[M]; p[0]=a*’ are both illegal C codes.

(3) You should not receive any compiler warnings like ‘incompatible pointer assignment’.

```
#include <stdio.h>
#include <stdlib.h>

int main()
{
   int **p = NULL;

   // Your codes here.

   // Now p should point to the matrix and the following code will be working.
   int i, j;
   for (i = 0; i < 10; ++i) {
      for (j = 0; j < 10; ++j)
        printf(“%d\t”, p[i][j]);
      printf(“\n”);
   }
   return 0;
}
```