CSCI-UA.0201-003

Computer Systems Organization

Midterm Exam - Fall 2014

(60 minutes)

Last name: First name:

Notes:

- **If you perceive any ambiguity in any of the questions, state your assumptions clearly**
- **Questions vary in difficulty; it is strongly recommended that you do not spend too much time on any one question.**
- **This exam is open book/notes.**

---

1. (4 points) Circle the correct answer among the choices given. If you circle more than one answer, you will lose the grade of the corresponding question.

(A) Registers usage convention is needed to:

1. make our programs correct
2. make our programs faster
3. make our programs work with other programs and library
4. make our programs more memory efficient

(B) Presenting +17 in signed integer or in IEEE 754 single precision floating point yields the same bit pattern.

1. The above statement is true
2. The above statement is false
3. It depends on whether the machine is 32-bit or 64-bit
4. It depends on whether the machine is big-endian or little endian.

(C) Which of the following has a larger size (in terms of bytes), assuming 32-bit machine?

1. pointer to a character
2. pointer to an integer
3. pointer to a float
4. pointer to pointer
5. They are all of the same size.

(D) Assume a function is called from two different places in a C program running on a 32-bit x86 machine:

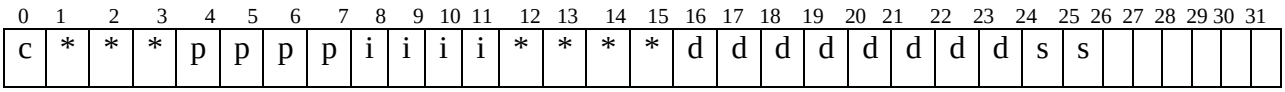
1. The stack frame of that function is of the same size each time.
2. The stack frame size of that function differs based on the arguments.

2. Given the following structure definition (assume 32-bit machine):

```
struct {
    char c;          double *p;          int i;
    double d;        short s;        } mystruct;
```

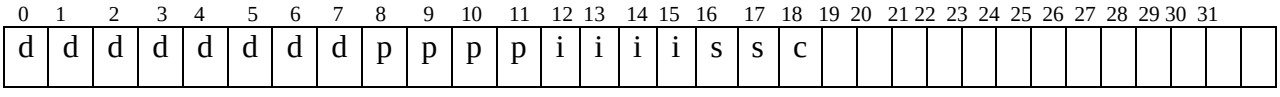
We said in class that a data item of size x must be stored in memory in an address multiple of x. We called this alignment.

a) (3 points) Fill out the following memory snapshot, showing how the structure items are placed in memory, including padding (if any). You start from address 0 (far left of the figure). You must show at which address every item starts and ends, where are the padding (if any), and the total number of bytes taken by the whole structure.



*: padding
blank: unused
Total: 26 bytes

b) (3 points) Can we re-order the items of the structure to take less space? If yes, fill the figure below showing the new order and the total space taken by the structure. If not, explain why.



You can exchange p and i and still get the best solution, total of 19 bytes.

3. (4 points) State two scenarios where (some) local variables of a function must be in the stack and not in registers?

Scenario 1: If we don't have enough registers to hold all local variables needed.

Scenario 2: If we want to generate an address for a variable (in case we want to assign it to a pointer for example)

4. For the following two functions (assume n >= 0):

```
int fact(int n)
{
  int i;
  int result = 1;

  for (i = n; i > 0; i--)
    result = result * i;

  return result;
}
```

```
int fact_u2(int n)
{
  int i;
  int result = 1;

  for (i = n; i > 0; i-=2) {
    result = (result * i) * (i-1);
  }

  return result;
}
```

(a) (2 points) What does the function **`fact`** do?

n! (i.e. factoriel n)

(b) (2 points) For which values of n do **`fact`** and **`fact_u2`** produce similar results?

For even values of n (including 0)

(c) (2 points) What is the minimum modification that we can do to **`fact_u2`** such that it always produces same result as **`fact`**?

Change the loop condition in fact_u2 to i > 1