

### Problem 1. (6 points):

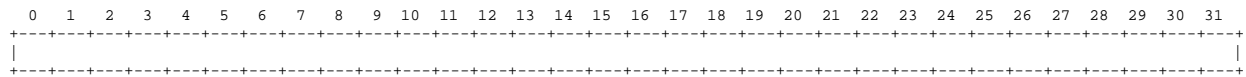
Consider the following datatype definitions on an IA32 (x86) machine.

```
typedef struct {
    char c;
    double *p;
    int i;
    double d;
    short s;
} struct1;

typedef union {
    char c;
    double *p;
    int i;
    double d;
    short s;
} union1;
```

A. Using the template below (allowing a maximum of 32 bytes), indicate the allocation of data for a structure of type `struct1`. Mark off and label the areas for each individual element (there are 5 of them). Cross hatch the parts that are allocated, but not used (to satisfy alignment).

Assume the alignment rules discussed in lecture: data types of size  $x$  must be aligned on  $x$ -byte boundaries. **Clearly indicate the right hand boundary of the data structure with a vertical line.**



B. How many bytes are allocated for an object of type `struct1`?

C. What alignment is required for an object of type `struct1`? (If an object must be aligned on an  $x$ -byte boundary, then your answer should be  $x$ .)

D. If we define the fields of `struct1` in a different order, we can reduce the number of bytes wasted by each variable of type `struct1`. What is the number of **unused, allocated** bytes in the best case?

E. How many bytes are allocated for an object of type `union1`?

F. What alignment is required for an object of type `union1`? (If an object must be aligned on an  $x$ -byte boundary, then your answer should be  $x$ .)