- CPSC 275: Introduction to Computer Systems

CPSC 275: Introduction to Computer Systems

Fall 2025

- Syllabus
- Schedule
- Resources
- Upload
- Solution

Homework 21

NOTE: You are not required to hand in the following exercises, but you are strongly encouraged to complete them to strengthen your understanding of the concepts covered in class.

1. Consider the following source code, where `M` and `N` are constants declared with `#define`:

   ```
   int mat1[M][N];
   int mat2[N][M];

   int sum_element(int i, int j) {
      return mat1[i][j] + mat2[j][i];
   }
   ```

   In compiling this program, `gcc` generates the following assembly code:

   ```
   # i at %ebp+8, j at %ebp+12
   movl 8(%ebp),%ecx
   movl 12(%ebp),%edx
   leal 0(,%ecx,8),%eax
   subl %ecx,%eax
   addl %edx,%eax
   leal (%edx,%edx,4),%edx
   addl %ecx,%edx
   movl mat1(,%eax,4),%eax
   addl mat2(,%edx,4),%eax
   ```

   Determine the values of `M` and `N` based on this assembly code.

2. Consider the following array, where *R*, *S*, and *T* are constants declared with `#define`:

   ```
   int A[R][S][T];
   ```

   provide a formula for the location of array element `A[i][j][k]`.

3. Given the following structure and variable definitions,

   ```
   struct customer {
      char lastName[15];
      char firstName[15];
      int customerNumber;
      struct {
         char phoneNumber[11];
         char address[50];
   ```

```
      char city[15];
      char state[3];
      char zipCode[6];
   } personal;
} customerRecord, *customerPtr;
customerPtr = &customerRecord;
```

write an expression that can be used to access the structure members in each of the following parts:

A. Member `lastName` of structure `customerRecord`.
B. Member `lastName` of the structure pointed to by `customerPtr`.
C. Member `firstName` of structure `customerRecord`.
D. Member `firstName` of the structure pointed to by `customerPtr`.
E. Member `customerNumber` of structure `customerRecord`.
F. Member `customerNumber` of the structure pointed to by `customerPtr`.
G. Member `phoneNumber` of member `personal` of structure `customerRecord`.
H. Member `phoneNumber` of member `personal` of the structure pointed to by `customerPtr`.
I. Member `address` of member `personal` of structure `customerRecord`.
J. Member `address` of member `personal` of the structure pointed to by `customerPtr`.
K. Member `city` of member `personal` of structure `customerRecord`.
L. Member `city` of member `personal` of the structure pointed to by `customerPtr`.
M. Member `state` of member `personal` of structure `customerRecord`.
N. Member `state` of member `personal` of the structure pointed to by `customerPtr`.
O. Member `zipCode` of member `personal` of structure `customerRecord`.
P. Member `zipCode` of member `personal` of the structure pointed to by `customerPtr`.

4. Consider the following structure declaration:

```
struct prob {
   int *p;
   struct {
      int x;
      int y;
   } s;
   struct prob *next;
};
```

This declaration illustrates that one structure can be embedded within another, just as arrays can be embedded within structures, and arrays can be embedded within arrays. The following procedure (with some expressions omitted) operates on this structure:

```
void sp_init(struct prob *sp)
{
   sp->s.x = ________;
   sp->p = ________;
   sp->next = ________;
}
```

A. What are the offsets (in bytes) of the following fields?

```
     p: ________
   s.x: ________
   s.y: ________
  next: ________
```

B. How many total bytes does the structure require?
C. The compiler generates the following assembly code for the body of `sp_init`:

```
# sp at %ebp+8
movl 8(%ebp), %eax
movl 8(%eax), %edx
movl %edx, 4(%eax)
leal 4(%eax), %edx
movl %edx, (%eax)
movl %eax, 12(%eax)
```

On the basis of this information, fill in the missing expressions in the code for `sp_init`.

5. Create union `integer` with members `char c`, `short s`, int `i` and long `b`. Write a program that inputs value of type `char`, `short`, `int` and `long` and stores the values in union variables of type union `integer`. Each union variable should be printed as a `char`, a `short`, an `int` and a `long`. Do the values always print correctly?

- **Welcome: Sean**
    - LogOut

Trinity College
HARTFORD CONNECTICUT