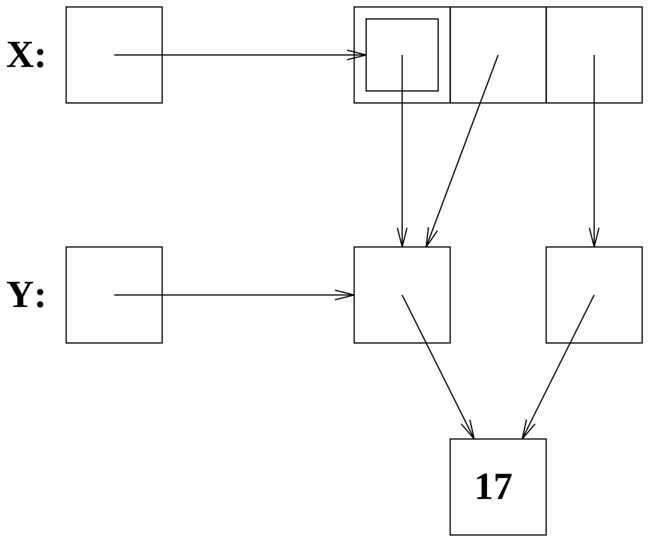
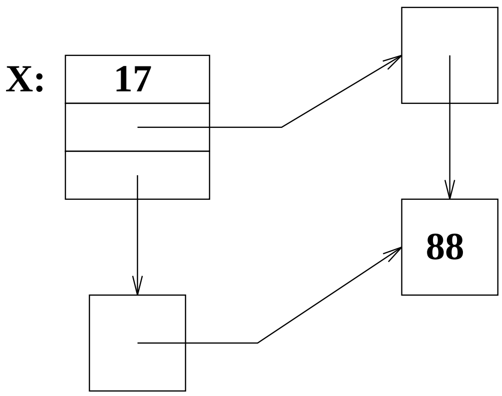
Assignment

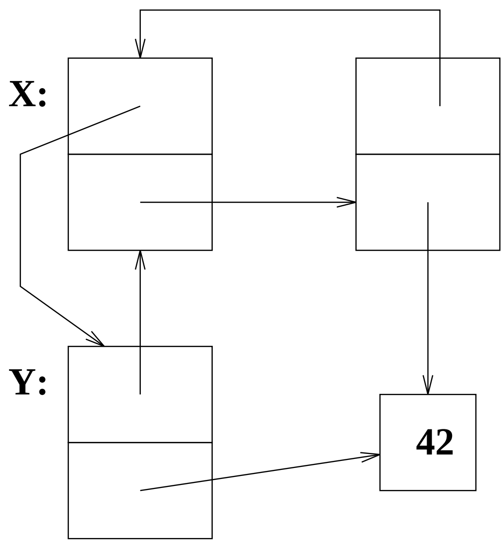
1. Write the type definitions, variable declarations, and code that will create the following picture:



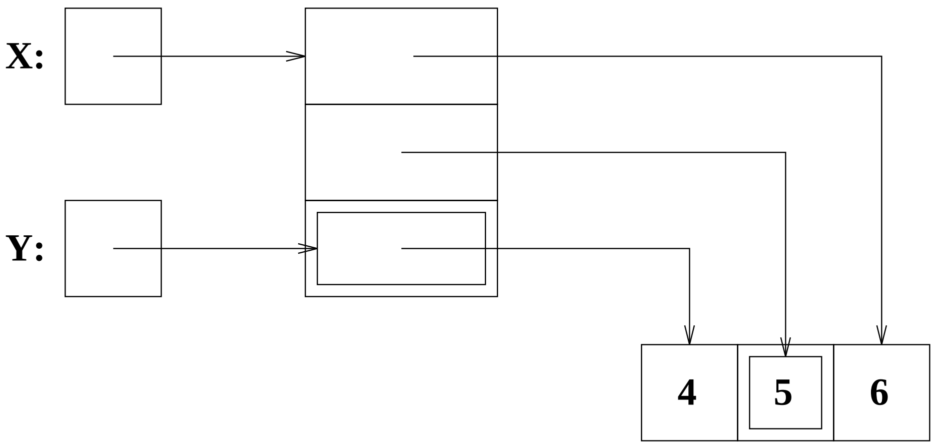
2. Write the type definitions, variable declarations, and code that will create the following picture:



3. Write the type definitions, variable declarations, and code that will create the following picture:



4. Write the type definitions, variable declarations, and code that will create the following picture:



5. Draw the results of the following code:

```
typedef int *int_ptr;
typedef int_ptr *int_ptr_ptr;
typedef struct {
  int_ptr_ptr field1;
  int_ptr_ptr field2;
} subproblem_struct;
typedef subproblem_struct *subprob_ptr;
typedef struct {
  subprob_ptr field1;
  subprob_ptr field2;
} problem_struct;

main()
{
  problem_struct problem;

  problem.field1 = malloc(sizeof(subproblem_struct));
  problem.field2 = malloc(sizeof(subproblem_struct));
  problem.field1->field1 = malloc(sizeof(int_ptr));
  problem.field1->field2 = problem.field1->field1;
  problem.field2->field1 = malloc(sizeof(int_ptr));
  problem.field2->field2 = problem.field2->field1;
  *problem.field1->field1 = malloc(sizeof(int));
  **problem.field1->field1 = 42;
  *problem.field2->field1 = malloc(sizeof(int));
  **problem.field1->field1 = 43;
}
```

6. Draw the results of the following code:

```
typedef int *int_ptr;
typedef int_ptr *int_ptr_ptr;
typedef struct {
  int_ptr_ptr field1;
  int_ptr_ptr field2;
} subproblem_struct;
typedef subproblem_struct *subprob_ptr;
typedef struct {
  subprob_ptr field1;
  subprob_ptr field2;
} problem_struct;

main()
{
  problem_struct problem;

  problem.field1 = malloc(sizeof(subproblem_struct));
  problem.field2 = malloc(sizeof(subproblem_struct));
  problem.field1->field1 = malloc(sizeof(int_ptr));
  problem.field1->field2 = malloc(sizeof(int_ptr));
  problem.field2->field1 = malloc(sizeof(int_ptr));
  problem.field2->field2 = malloc(sizeof(int_ptr));
  *problem.field1->field1 = malloc(sizeof(int));
  *problem.field1->field2 = *problem.field1->field1;
  **problem.field1->field1 = 42;
  *problem.field2->field1 = malloc(sizeof(int));
  *problem.field2->field2 = *problem.field2->field1;
  **problem.field2->field1 = 43;
}
```

7. Draw the results of the following code:

```
typedef int *int_ptr;
typedef int_ptr *int_ptr_ptr;
typedef struct {
  int_ptr_ptr field1;
  int_ptr_ptr field2;
} subproblem_struct;
typedef subproblem_struct *subprob_ptr;
typedef struct {
  subprob_ptr field1;
  subprob_ptr field2;
} problem_struct;

main()
{
  problem_struct problem;

  problem.field1 = malloc(sizeof(subproblem_struct));
  problem.field2 = malloc(sizeof(subproblem_struct));
  problem.field1->field1 = malloc(sizeof(int_ptr));
  problem.field1->field2 = malloc(sizeof(int_ptr));
  problem.field2->field1 = malloc(sizeof(int_ptr));
  problem.field2->field2 = malloc(sizeof(int_ptr));
  *problem.field1->field1 = malloc(sizeof(int));
  *problem.field1->field2 = malloc(sizeof(int));
  *problem.field2->field1 = malloc(sizeof(int));
  *problem.field2->field2 = malloc(sizeof(int));
  **problem.field1->field1 = 73;
  **problem.field1->field2 = 42;
  **problem.field2->field1 = 74;
  **problem.field2->field2 = 43;
}
```

8. Draw the results of the following code:

```
typedef int *int_ptr;
typedef struct problem_struct *pstruct_ptr;
typedef pstruct_ptr prob_array[2];
typedef prob_array *p_array_ptr;
typedef struct problem_struct {
  int_ptr field1;
  p_array_ptr field2;
} p_struct_type;

main()
{
  prob_array A;

  A[0] = malloc(sizeof(p_struct_type));
  A[1] = malloc(sizeof(p_struct_type));
  A[0]->field1 = malloc(sizeof(int));
  A[1]->field1 = A[0]->field1;
  *A[0]->field1 = 2001;
  A[0]->field2 = &A;
  A[1]->field2 = A[0]->field2;
}
```

Submission

Note that the usual "midnight rule" does not apply for this assignment; to be on time you must submit your assignment by the *start* of class. No submission will be accepted after the start of class two days after the due date.

You may submit your pictures handwritten on paper. Your code must be submitted *typewritten* on paper. You may find it helpful to compile and run each code sequence you write. Doing so will not guarantee that your answers are correct, but at least it will reassure you that you do not have any glaring errors. Please place your name, student ID number, and comment about help received at the top of your submission.