

# ECE264 Fall 2021 Exam 1

**6:30-7:30PM, September 30**

Please keep the answer sheet clean. Do not use the answer sheet as your scratch space. The answer sheet should contain **only** the answers.

Do not write anything that is not the answers.  
Otherwise, you may lose points.

Please use **DARK** ink. If your pen is too light, your answer may not be graded.

When you take this exam, you agree with the Purdue Honor Pledge:  
**advancing a culture of academic integrity, seeing academic dishonesty and cheating as threats to Purdues reputation.**

| Value | Character | Value | Character | Value | Character |
|-------|-----------|-------|-----------|-------|-----------|
| 48    | 0         | 65    | A         | 97    | a         |
| 49    | 1         | 66    | B         | 98    | b         |
| 50    | 2         | 67    | C         | 99    | c         |
| 51    | 3         | 68    | D         | 100   | d         |
| 52    | 4         | 69    | E         | 101   | e         |
| 53    | 5         | 70    | F         | 102   | f         |
| 54    | 6         | 71    | G         | 103   | g         |
| 55    | 7         | 72    | H         | 104   | h         |
| 56    | 8         | 73    | I         | 105   | i         |
| 57    | 9         | 74    | J         | 106   | j         |
|       |           | 75    | K         | 107   | k         |
|       |           | 76    | L         | 108   | l         |
|       |           | 77    | M         | 109   | m         |
|       |           | 78    | N         | 110   | n         |
|       |           | 79    | O         | 111   | o         |
|       |           | 80    | P         | 112   | p         |
|       |           | 81    | Q         | 113   | q         |
|       |           | 82    | R         | 114   | r         |
|       |           | 83    | S         | 115   | s         |
|       |           | 84    | T         | 116   | t         |
|       |           | 85    | U         | 117   | u         |
|       |           | 86    | V         | 118   | v         |
|       |           | 87    | W         | 119   | w         |
|       |           | 88    | X         | 120   | x         |
|       |           | 89    | Y         | 121   | y         |
|       |           | 90    | Z         | 122   | z         |

# 1 Pointers and Arrays

Consider the following piece of C code:

```
1 #include <stdlib.h>
2 #include <stdio.h>
3
4 int main(int argc, char * argv[])
5 {
6     int * x = malloc(5 * sizeof(int));
7     int * y = malloc(5 * sizeof(int));
8     int * * p = &x;
9     int * * q = &y;
10
11     for (int i = 0; i < 5; i++) {
12         x[i] = i;          //x has values [0, 1, 2, 3, 4]
13         y[i] = 4 - i;      //y has values [4, 3, 2, 1, 0]
14     }
15
16     printf("* * p = %d\n", * * p); // <=== ANSWER A
17
18     printf("* (*q + 1) = %d\n", * (*q + 1)); // <=== ANSWER B
19
20     * q = (*p + 1);
21     printf("y[1] = %d\n", y[1]); // <=== ANSWER C
22
23     * p = (*q + 1);
24     printf("x[1] = %d\n", x[1]); // <=== ANSWER D
25
26     return EXIT_SUCCESS;
27 }
```

Each print statement prints out an integer value. The answers to this question are those integer values

Hint: The output of the program will be

```
* * p = [ANSWER A]
* (*q + 1) = [ANSWER B]
y[1] = [ANSWER C]
x[1] = [ANSWER D]
```

## 2 Structures

Consider the following structure definitions:

```
1 typedef struct {
2     int p[10];
3     int q[10];
4 } A;
5
6 typedef struct {
7     int r[10];
8     int s[10];
9 } B;
10
11 typedef struct {
12     int x;
13     A * y;
14     B * z;
15 } C;
```

### ANSWER A:

Assume you have a structure variable `var`:

`C var`;

How would you allocate space for the `y` field of `var`? Your answer should be one line of C code.

### ANSWER B:

Assume that all of the fields of `var` are correctly allocated and initialized.

Which of these statements makes the 3rd element of the second field of the `z` field of `var` equal to 7? Your answer should be the number of the correct statement from the list below.

1. `var.z.s[3] = 7;`
2. `var.z->s[2] = 7;`
3. `var->z.s[3] = 7;`
4. `var->z.s[2] = 7;`
5. `var.z.s[2] = 7;`
6. `var.z->s[3] = 7;`

### 3 Makefile

Consider the following Makefile and answer the questions A - D.

```
1 CFLAGS = -std=c99 -g -Wall -Wshadow --pedantic -Wvla -Werror
2 GCC = gcc $(CFLAGS)
3 EXEC = q3prog
4 TESTFLAGS = -DTEST_COMPARE
5
6 SRCS = add.c compare.c main.c
7 OBJS = $(SRCS:%.c=%.o)
8 # <--- Question A: What is the value of OBJS? --->
9
10 all: $(EXEC)
11
12 # link .o to executable
13 $(EXEC): $(OBJS)
14     # <--- Question B: Fix the following --->
15     $(GCC) $(TESTFLAGS) $(OBJS)      [ANSWER B]      $(EXEC)
16
17 # convert .c to .o
18 %.o: %.c
19     $(GCC) $(TESTFLAGS) -c $<
20
21 testall: test1 test2 test3
22
23 test1: $(EXEC)
24     ./$(EXEC) inputs/input1 > output1
25
26 test2: $(EXEC)
27     ./$(EXEC) inputs/input2 > output2
28
29 test3: $(EXEC)
30     ./$(EXEC) inputs/input3 > output3
31
32 clean: # remove all machine generated files
33     rm -f $(EXEC) *.o *.out *gcda *gcno *gcov
```

**ANSWER A:** Write down the value of OBJS.

**ANSWER B:** Fix the command at line 15. Your answer should be what needs to go in [ANSWER B]

**ANSWER C:** Using the given **Makefile**, what is the Unix command to test with all test cases (input1, input2, input3)? Your answer should be a single command (plus any needed arguments) you can type at the command line.

**ANSWER D:** Suppose we have the following in **compare.c**:

```
# ifdef TEST_COMPARE
void func1(void) {
    ...
}
# endif

# ifdef TEST_FUNC
void func2(void) {
    ...
}
# endif
```

When we use the given **Makefile** to test the program, which function(s) is/are compiled and tested? Your answer should be the number of the correct response.

1. func1
2. func2
3. Both func1 and func2
4. None of them

## 4 Memory Leak

Consider the following function:

```
1 #include <stdio.h>
2 #include <stdlib.h>
3 #include <string.h>
4 #pragma pack(1) // tell compiler not to pad any space
5 // assume sizeof(char) = 1, sizeof(a pointer) = 8
6
7 int main(int argc, char ** argv)
8 {
9     if (argc < 2) {
10         printf("Not enough inputs\n");
11         return EXIT_FAILURE;
12     }
13
14     char ** arr = malloc((argc-1) * sizeof(char *));
15     int i = 0;
16     for(; i < argc-1; i++) {
17         // strlen(argv[i+1]) computes the length of the
18         // string argv[i+1], not including the
19         // terminating null character, '\0'.
20         arr[i] = malloc((strlen(argv[i+1]) + 1) * sizeof(char));
21
22         // strcpy copies the string pointed to
23         // by argv[i+1] to arr[i]
24         strcpy(arr[i], argv[i+1]);
25         printf("%s\n", arr[i]);
26     }
27
28     for(i = 0; i < argc-1; i++) {
29         free(arr[i]); // release memory
30     }
31     // <--- Question A: Fix the following code to release memory --->
32     free(    [Question A]    );
33
34     return EXIT_SUCCESS;
35 }
```

Suppose the executable program is named q4prog.

We run the program with

```
./q4prog ece 264 midterm1
```

Answer the questions A - C.

Hint: The output of the program will be

```
ece  
264  
midterm1
```

**ANSWER A:** Fix the code at line 32 to properly release memory. Your answer should be the expression that is the argument to **free**.

**ANSWER B:** If we do not release memory at line 32, how many bytes of memory will be leaked?

**ANSWER C:** If we do not release memory at line 29, but we *do* release the memory at line 32, how many bytes of memory will be leaked?



## 5 Data Types

Consider the following program that reads an input file as integers or floating point numbers.

```
1 #include <stdio.h>
2 #include <stdlib.h>
3
4 int main(int argc, char * * argv)
5 {
6     if (argc != 2)
7     {
8         fprintf(stderr, "Need one input file\n");
9         return EXIT_FAILURE;
10    }
11    FILE * fptr = fopen(argv[1], "r");
12    if (fptr == NULL) // fopen fail
13    {
14        return EXIT_FAILURE;
15    }
16    int count = 0;
17    // -----
18    // treat the input as integers
19    count = 0;
20    fseek(fptr, 0, SEEK_SET); // return to the beginning of the file
21    int ival;
22    while (fscanf(fptr, "%d", & ival) == 1)
23    {
24        count ++;
25    }
26    printf("The file has %d integers.\n", count); // <=== ANSWER A
27    printf("The last read integer is %d.\n", ival); // <=== ANSWER B
28    // -----
29    // treat the input as float
30    count = 0;
31    fseek(fptr, 0, SEEK_SET); // return to the beginning of the file
32    float fval;
33    while (fscanf(fptr, "%f", & fval) == 1)
34    {
35        count ++;
36    }
37    printf("The file has %d float.\n", count); // <=== ANSWER C
38    printf("The last floating-point is %f.\n", fval); // <=== ANSWER D
39    // -----
```

```

40 // understand the sizes of different types
41 // assume
42 // sizeof(char) is 1
43 // sizeof(int) is 4
44 // sizeof(int*) is 8
45 // sizeof(double) is 8
46 double darr[] = {2.64, 20.11, 0.9, 2.8};
47 int sizediff = (int)sizeof(darr) - (int)sizeof(darr[0]);
48 printf("%d\n", sizediff); // <== ANSWER E
49 return EXIT_SUCCESS;
50 }

```

This is the input file:

```
2 64 20.11 9 17
```

The following table shows bytes of this input file.

|                 |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |
|-----------------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|
| Byte            | 0  | 1  | 2  | 3  | 4  | 5  | 6  | 7  | 8  | 9  | 10 | 11 | 12 | 13 | 14 |
| Character       | 2  |    | 6  | 4  |    | 2  | 0  | .  | 1  | 1  |    | 9  |    | 1  | 7  |
| ASCII (decimal) | 50 | 32 | 54 | 52 | 32 | 50 | 48 | 46 | 49 | 49 | 32 | 57 | 32 | 49 | 55 |

Suppose you run the program with the above input file. Please write down the program's output from the five print statements. **Your answers should be ONLY the numbers.**