

ECE264 Summer 2019

Exam 1, 8:40-9:40AM, June 26th

In signing this statement, I hereby certify that the work on this exam is my own and that I have not copied the work of any other student while completing it. I understand that, if I fail to honor this agreement, I will receive a score of ZERO for this exam and will be subject to possible disciplinary action.

Purdue username:

Signature:

*You must sign here. Otherwise you will receive a **1-point** penalty.*

Read the questions carefully.
Some questions have conditions and restrictions.

This is an *open-book, open-note* exam. You may use any book, notes, or program printouts. No personal electronic device is allowed. You may **not** borrow books from other students.

Contents

1	Conditional compilation and Makefile (10 points)	3
2	Structure (40 points)	4
3	Comparison Function for qsort (20 points)	6
4	Pointer (30 points)	8
5	Errata	9

Total Score:

1 Conditional compilation and Makefile (10 points)

1. **(5 points)** Disney Pictures released the latest version (hint: version 4) of the movie Toy Story recently. An identifier `VERSION` in the below code is defined based on the latest version of the movie released. The code is compiled using the command:

```
gcc -Wall -Werror -Wshadow -Wvla --pedantic -g -std=c99 -o toystory
```

and run using the command:

```
./toystory 4
```

```
1  #include<stdio.h>
2  #include<stdlib.h>
3
4  int main(int argc, char** argv)
5  {
6      int VERSION=atoi(argv[1]);
7      #if VERSION > 3
8          printf("Toy Story %d\n", VERSION);
9      #elif defined(VERSION3)
10         printf("Toy Story 3\n");
11     #else
12         printf("No Story\n");
13     #endif
14         return 0;
15 }
```

What is the output printed? Please write below:

2. **(5 points)** The above code is saved in a file called `toystory.c`. Update the contents of a makefile shown below to create a target named `toystory` from the source file `toystory.c`.

```
GCC = gcc -Wall -Werror -Wshadow -Wvla --pedantic -g -std=c99
FLAGS = -DVERSION3
        : toystory.c
        $(GCC) $(FLAGS)                -o
```

What is the output printed when you use the same run command as in part 1? Please write below:

2 Structure (40 points)

Each question is 10 points.

Consider the following program:

```
1 #include <stdio.h>
2 #include <stdlib.h>
3 #pragma pack(1) // tell compiler not to pad any space
4 // assume sizeof(char) = 1, sizeof(int) = 4,
5 // sizeof(double) = 8, sizeof(a pointer) = 8
6 #define NAME_LENGTH 20
7 #define NUM_STUDENT 10
8 typedef struct
9 {
10     int ID;
11     char * firstName;
12     char lastName [NAME_LENGTH];
13 } Student;
14
15 void allocateStudent(Student * * ptraddr, int number)
16 {
17     Student * ptr = malloc(sizeof(Student) * number);
18     * ptraddr = ptr;
19 }
20
21 void allocateName(char * * ptraddr, int number)
22 {
23     char * ptr = malloc(sizeof(char) * number);
24     * ptraddr = ptr;
25 }
26
27 int main(int argc, char ** argv)
28 {
29     Student * sptr;
30     printf("sizeof(sptr)      = %ld\n", sizeof(sptr));
31     printf("sizeof(* sptr) = %ld\n", sizeof(* sptr));
32
33     allocateStudent(& sptr, NUM_STUDENT);
34     // assume the output of
35     printf("& sptr[1] = %p\n", (void *) & sptr[1]);
36     // is ---> 0x050
37
38     // what is the output of the following statement?
```

```

39 // please write in hexadecimal (answer A)
40 printf("& sptr[0] = %p\n", (void *) & sptr[0]);
41
42 // what is the output of the following statement?
43 // please write in hexadecimal (answer B)
44 printf("& sptr[2] = %p\n", (void *) & sptr[2]);
45
46 allocateName(& sptr[0].firstName, NAME_LENGTH);
47
48 // assume the output of
49 printf("& sptr[0].firstName[0] = %p\n",
50        (void *) & sptr[0].firstName[0]);
51 // is ---> 0xa30
52
53 // what is the output of the following statement?
54 // please write in hexadecimal (answer C)
55 printf("& sptr[0].firstName[9] = %p\n",
56        (void *) & sptr[0].firstName[9]);
57
58 sptr[0].lastName[32] = 'E';
59 // please notice that the index is too large
60 sptr[0].lastName[33] = 'C';
61 sptr[0].lastName[34] = 'E';
62 sptr[0].lastName[35] = '2';
63 sptr[0].lastName[36] = '6';
64 sptr[0].lastName[37] = '4';
65 sptr[0].lastName[38] = '\0';
66
67 // What is the output of this statement? (answer D)
68 printf("%s\n", & sptr[1].lastName[2]);
69
70 free(sptr[0].firstName);
71
72 free(sptr);
73 return EXIT_SUCCESS;
74 }

```

3 Comparison Function for qsort (20 points)

Each answer is 5 points.

```
1 #include <stdio.h>
2 #include <stdlib.h>
3 #include <stdarg.h>
4 #define NAME_LENGTH 80
5 #define NUM_PERSONS 10
6 typedef struct
7 {
8     char lastName[NAME_LENGTH];
9     char firstName[NAME_LENGTH];
10    char address[NAME_LENGTH];
11 } Person;
12
13 // comparsion function by last name, first name, address
14 // This comparison function is used by qsort
15 // use strcmp to compare the strings
16 int comparePerson(const void * p1,  const void * p2)
17 {
18     // step 1: convert p1 and p2 into pointers to Persons
19     // Answer A:
20
21
22
23
24
25
26
27
28
29     // step 2: compare the attributes
30     // compare the last names
31     // Answer B:
32
33
34
35
36
37
38
39
```

```

40
41
42
43 // if the last names are the same, compare the first names
44 // Answer C:
45
46
47
48
49
50
51
52
53
54
55 // if the first names are the same, compare the addresses
56 // Answer D:
57
58
59
60
61
62
63
64
65
66
67 }
68
69 int main (int argc, char * * argv)
70 {
71 //Allocate array of Persons
72 Person * arr = malloc(NUM_PERSONS * sizeof(Person));
73 //Read array data in from a file (not shown)
74 readRecords(arr, NUM_PERSONS, argv[1]);
75 //Sort array using qsort
76 qsort(arr, NUM_PERSONS, sizeof(Person), comparePerson);
77 //Write sorted array data out to a file (not shown)
78 writeRecords(arr, NUM_PERSONS, argv[2]);
79 }

```

4 Pointer (30 points)

Write the outputs of this program.

Each pair of answer is 6 points.

```
1 #include <stdio.h>
2 #include <stdlib.h>
3 typedef void (* swapptr)(int *, int *);
4
5 void swap1(int * a, int * b)
6 {
7     int * t;
8     t = a;
9     a = b;
10    b = t;
11 }
12
13 void swap2(int * a, int * b)
14 {
15     int * s;
16     s = a;
17     * a = * b;
18     b = s;
19 }
20
21 void swap3(int * a, int * b)
22 {
23     int s;
24     s = * a;
25     a = b;
26     * b = s;
27 }
28
29 void swap4(int * a, int * b)
30 {
31     int * s;
32     s = a;
33     * a = * b;
34     * b = * s;
35 }
36
37
38
```



```

39
40
41
42 void swap5(int * a, int * b)
43 {
44     int s;
45     s = * a;
46     * a = * b;
47     * b = s;
48 }
49
50 int main(int argc, char * * argv)
51 {
52     swapptr swaparray [] = {swap1, swap2, swap3, swap4, swap5};
53     int numswap = sizeof(swaparray) / sizeof(swapptr);
54     for (int ind = 0; ind < numswap; ind ++)
55     {
56         swapptr func = swaparray[ind];
57         int s = 2;
58         int t = 6;
59         func(& s, & t);
60         printf("%d %d\n", s, t);
61     }
62     return EXIT_SUCCESS;
63 }

```

5 Errata

q1, part 1 the correct command to compile is: `gcc q1.c -o toystory`