

CS 354 (Spring 2019) @ Epic

Worksheet 3

Total points: 28

Due: Feb 19th (Tue) before 11:59 pm via canvas.wisc.edu

1. Continuing from the method `reverse_num()` in the last worksheet, assume that instead of an integer, input to `reverse_num()` is passed as a string. Your function should return the reversed number in the form of a string instead of an integer. But you need to handle preceding zeros. For example, on passing "743", "347" should be returned. But on passing "00743" and "74300", "34700" and "00347" should be returned respectively. Your function should also check if the input string is a valid integer. If not, `reverse_num()` should return NULL. Complete the following methods: **[10 points]**

```
// Returns 1 if num is a valid integer, otherwise returns 0
int is_valid_num(char *num) {
```

```
}
```

```
// num: Integer to be reversed
char* reverse_num(char *num) {
```

```
}
```

2. Let's assume an array as shown below:

```
int a[2] = {1, 2};  
int *iptr;  
iptr = a;
```

```
char b[2];  
char *cptr;  
cptr = b;
```

Given: Array a starts at address 0x100, b starts at address 0x200. Assume that an integer takes 4 bytes of memory and a character takes 1 byte of memory. Fill the values for the following expressions. In other words, what is the value that will be printed if these expressions are used in a printf(). e.g., printf("%p", iptr); or printf("%d", *iptr); You may assume that each print expression is evaluated independently. In other words, the values of the previous expressions (if any) do not have any effect on the value of the current expression. If the value of any expression is indeterminate, you should write "indeterminate". If any expression is illegal in C, you should write "illegal". **[10 points]**

Expression	Value
iptr	
*iptr	
iptr + 1	
*(iptr + 2)	
(iptr + 1) - (iptr)	
*cptr	
cptr + 1	
(cptr + 1) - (cptr)	
(char *) iptr + 1	
iptr + cptr	

3. Let's say you have the following structure array for the 50 students in CS 354:

```
struct studinfo {  
    char *name;  
    unsigned int age;  
};
```

```
struct studinfo student[50];  
struct studinfo *studentptr = student;
```

Choose the correct way(s) in which you can access the name of the student at index 2 (i.e., student[2]). **[4 points]**

- i. student[2].name
- ii. (studentptr + 2)->name
- iii. *(studentptr + 2)->name
- iv. student[2]->name
- v. studentptr[2].name
- vi. studentptr[2]->name
- vii. *(student + 2).name
- viii. *(student + 2)->name

Note: You only have to **underline** the correct way(s) here. For each incorrect way that is underlined, 1 point will be deducted! You need not give any reason for your answer.

4. Consider a method foo. It accepts as parameter, a 2D array with 5 rows and 10 columns and returns void. Which of the following is/are valid function signature(s) for foo? **Underline** the correct function signature(s) here. Also, state the reason for your answer. **[4 points]**

- i. void foo(int arr[][]);
- ii. void foo(int arr[][10]);
- iii. void foo(int arr[5][]);
- iv. void foo(int arr[5][10]);

Reason: