Question 1

0 out of 0 points

Please type your full name below, whereby you pledge on your honor that you will neither give nor receive any unauthorized assistance on this examination.

Selected Answer:	Alan Biju Palayil
Correct Answer:	[None]

Question 2

3 out of 3 points

Consider the following C declarations:

```
int iarr[100];
void *p = iarr;
```

Which of the following expressions is semantically equivalent to "`iarr[50]`"

Selected Answer:	✓ `((int *)((char *)p + 50 * sizeof(int)))[0]`
Correct Answer:	✓ `((int *)((char *)p + 50 * sizeof(int)))[0]`

Question 3

3 out of 3 points

Which of the following is *false* regarding procedure calls in C?

Selected Answer:	✓ primitive types (e.g., `int`, `char`, `float`, etc.) are passed by value, but composite types (e.g., `structs`) are passed by reference
Correct Answer:	✓ primitive types (e.g., `int`, `char`, `float`, etc.) are passed by value, but composite types (e.g., `structs`) are passed by reference

Question 4

3 out of 3 points

At which stage of the compilation process are macro invocations replaced with their definitions?

Selected Answer:	✓ preprocessing
Correct Answer:	✓ preprocessing

Question 5

3 out of 3 points

What class of exceptions does a system call belong to?

Selected Answer:	✓ traps
Correct Answer:	✓ traps

Question 6

3 out of 3 points

Which of the following is not an example of a synchronous exception?

Selected Answer:	✓ a keystroke (e.g., ctrl-C)
Correct Answer:	✓ a keystroke (e.g., ctrl-C)

Question 7 3 out of 3 points

Which of the following is inherited by a child process from its parent when forking?

Selected Answer: ✓ blocked signals

Correct Answer: ✓ blocked signals

Question 8 3 out of 3 points

Which category of process does the kernel always take over responsibility for reaping?

Selected Answer: ✓ those whose parents have already terminated

Correct Answer: ✓ those whose parents have already terminated

Question 9 3 out of 3 points

Consider a scenario where a process running program A performs an exec in order to run program B. What happens when program B terminates the process by calling exit?

Selected Answer: ✓ the process will turn into a zombie

Correct Answer: ✓ the process will turn into a zombie

Question 10 3 out of 3 points

Consider a scenario where a long-running foreground job has been started from a shell and the user interrupts the job by hitting Ctrl-C at the terminal. Which best describes the events that lead to the job's termination?

Selected Answer: ✓ The terminal emulator sends a SIGINT signal to the shell; the shell forwards the signal onto the foreground job, which in turn terminates when the signal is delivered.

Correct Answer: ✓ The terminal emulator sends a SIGINT signal to the shell; the shell forwards the signal onto the foreground job, which in turn terminates when the signal is delivered.

Question 11 0 out of 3 points

Which of the following statements about signal delivery is *true* by default?

Selected Answer: ✗ when a process receives the `SIGINT` signal, it is also delivered to all of its children

Correct Answer: ✓ while a given signal is being handled, it will not be delivered again

Question 12 3 out of 3 points

Which of the following actions is most likely to cause the containing function to be *non-reentrant*?

Selected Answer: ✓ modifying a global data structure

Correct Answer: ✓ modifying a global data structure

Question 13

3 out of 3 points

Consider a scenario where function **f** has been registered as the handler for two different signals **S1** and **S2**, where **S1** has a higher priority than **S2**

Given that signal **S1** has just been delivered and **f** is currently executing, what happens if **S2** were to arrive?

Selected Answer:	✓ **S2** is marked as pending but is not delivered
Correct Answer:	✓ **S2** is marked as pending but is not delivered

Question 14

3 out of 3 points

In your implementation of a shell, you invoked the `waitpid` system call in a loop within the `SIGCHLD` signal handler. Why was this important?

Selected Answer:	✓ multiple child processes terminating could result in only one `SIGCHLD` being delivered
Correct Answer:	✓ multiple child processes terminating could result in only one `SIGCHLD` being delivered

Question 15

0.66666 out of 4 points

Place the following actions in the order that they would take place from the time signal **S** is sent to the time its delivery to the destination process is complete.

Correct Answer	Selected Answer
✓ 1. Signal **S** is marked as pending in the `pending` bit vector	✓ 1. Signal **S** is marked as pending in the `pending` bit vector
✓ 2. The kernel computes the value `pending & ~blocked`	✗ 2. Signal **S** is marked as blocked
✓ 3. Signal **S** is marked as blocked	✗ 3. The kernel computes the value `pending & ~blocked`
✓ 4. The kernel calls the registered handler for signal **S**	✗ 4. Signal **S** is unblocked
✓ 5. The handler for signal **S** completes.	✗ 5. The kernel calls the registered handler for signal **S**
✓ 6. Signal **S** is unblocked	✗ 6. The handler for signal **S** completes.

Question 16

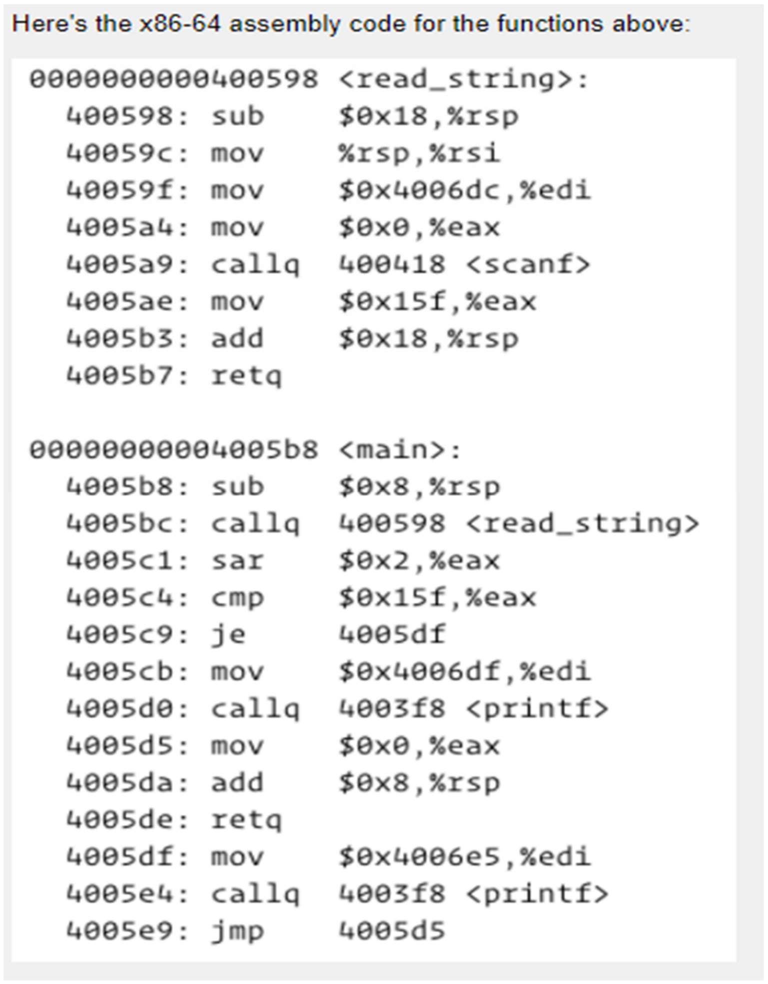
0 out of 8 points

This problem concerns the following C code, for which you are to design a buffer overflow attack such that when it is run the program will print "You've defused me!".

```c
int read_string() {
  char buf[5];
  scanf("%s", buf);
  return 351;
}

int main() {
  int val = read_string();
  val >>= 2;
  if (val != 351)
    printf("Boom!");
  else
    printf("You've defused me!");
  return 0;
}
```

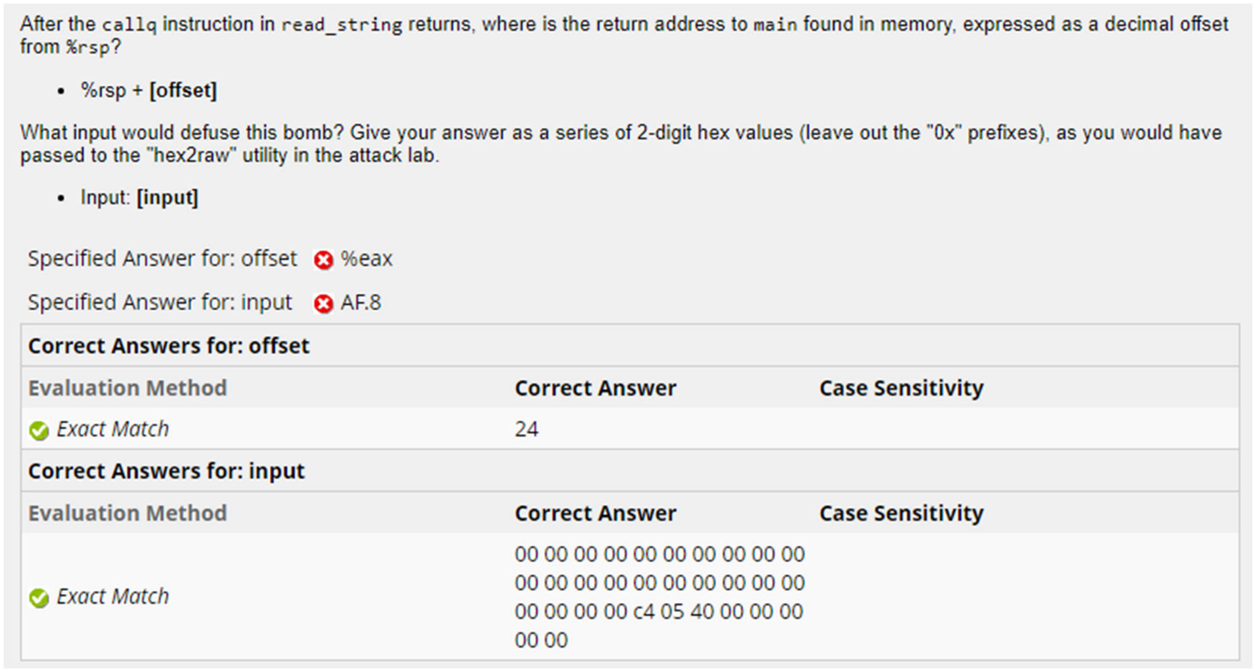
Note that `scanf("%s", buf)` reads an input string from stdin and stores it at address `buf` (including the terminating `'\0'` character). It does not check the size of the destination buffer.



Here's the x86-64 assembly code for the functions above:

```
0000000000400598 <read_string>:
  400598: sub    $0x18,%rsp
  40059c: mov    %rsp,%rsi
  40059f: mov    $0x4006dc,%edi
  4005a4: mov    $0x0,%eax
  4005a9: callq  400418 <scanf>
  4005ae: mov    $0x15f,%eax
  4005b3: add    $0x18,%rsp
  4005b7: retq

00000000004005b8 <main>:
  4005b8: sub    $0x8,%rsp
  4005bc: callq  400598 <read_string>
  4005c1: sar    $0x2,%eax
  4005c4: cmp    $0x15f,%eax
  4005c9: je     4005df
  4005cb: mov    $0x4006df,%edi
  4005d0: callq  4003f8 <printf>
  4005d5: mov    $0x0,%eax
  4005da: add    $0x8,%rsp
  4005de: retq
  4005df: mov    $0x4006e5,%edi
  4005e4: callq  4003f8 <printf>
  4005e9: jmp    4005d5
```



After the `callq` instruction in `read_string` returns, where is the return address to `main` found in memory, expressed as a decimal offset from `%rsp`?

- %rsp + **[offset]**

What input would defuse this bomb? Give your answer as a series of 2-digit hex values (leave out the "0x" prefixes), as you would have passed to the "hex2raw" utility in the attack lab.

- Input: **[input]**

Specified Answer for: offset ✗ %eax

Specified Answer for: input ✗ AF.8

Correct Answers for: offset		
Evaluation Method	**Correct Answer**	**Case Sensitivity**
✓ *Exact Match*	24	

Correct Answers for: input		
Evaluation Method	**Correct Answer**	**Case Sensitivity**
✓ *Exact Match*	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 c4 05 40 00 00 00 00 00	

Question 17

4 out of 8 points

For the following program:

1. Sketch the corresponding process tree --- being sure to indicate outputs and circle synchronization points, if they exist.
2. List *all* distinct outputs that could be produced when it is executed.

```
main() {
  int x = 0;
  printf("1");
  if (fork() == 0) {
    printf("2");
    x += 1;
  } else if (fork() == 0) {
    printf("3");
    x += 1;
  }
  if (x < 1) {
    while (wait(NULL) > 0) ;
    printf("4");
  } else {
    printf("5");
  }
}
```

Upload an image file (JPG, PNG, or PDF only!) containing both parts of your solution. You may either hand write your solution and upload a photo or use graphics software.

Selected Answer: Midterm Q17.jpg

Question 18

8 out of 8 points

For the following program:

1. Sketch the corresponding process tree --- being sure to indicate outputs and circle synchronization points, if they exist.
2. List *all* distinct outputs that could be produced when it is executed.

```
main() {
  printf("1");
  if (fork() == 0) {
    for (int i=2; i<4; i++) {
      printf("%d", i);
      if (fork() == 0) {
        printf("%d", i+10);
        exit(0);
      }
      wait(NULL);
    }
  } else {
    wait(NULL);
    printf("4");
  }
}
```

Upload an image file (JPG, PNG, or PDF only!) containing both parts of your solution. You may either hand write your solution and upload a photo or use graphics software.

Question 19

2 out of 8 points

Consider the following program:

```
int arr[100] = { 0 };
int idx = 0;
int val = 10;

void sighandler(int sig) {
  arr[idx] = val;
  idx++;
  val--;
}

main() {
  signal(SIGUSR1, sighandler);
  signal(SIGUSR2, sighandler);

  if (fork() == 0) {
    for (int i=0; i<10; i++) {
      kill(getppid(), SIGUSR2);
      kill(getppid(), SIGUSR1);
    }
    exit(0);
  } else {
    wait(NULL);
    for (int i=0; i<10; i++) {
      printf("%d ", arr[i]);
    }
  }
}
```

Occasionally, the program behaves even more oddly, and produces output like this:

```
10 10 9 8 6 6 5 3 2 1
```

Explain how this output might be produced. Be specific.

Selected Answer: In the operation, firstly we initialize array with 0, and create userdefine method sighandler which assign array elements its value ranging from 10, 0. SIGUSR1 and SIGUSR2 go through the else loop and print the values 11, 22, 33 and so on. Once in the if loop we kill the SIGUSR to terminate the process forcefully. The reason we get different outputs is mainly because the print function is not keeping up with the calculation speed and in between the for loop if the signal is available then the sighandler is called by kill function explicitly producing the given output.

Correct Answer: [None]