Question 1.

```
unsigned char a = 255;
signed char b = FFH;
signed char c = 80H;
unsigned char d = 0;
a = a+1;
b = b+1;
c = -c;
d = c
```

After executing the above code segment, what will be the final values of the variables **in decimal**?

Question 2.

```
int main(int argc, char *argv[]){
   char buf [100];

   sprintf(buf, "grep %s /etc/passwd",argv[1]);
   printf (buf);
   system(buf);
   exit(0);
}
```

Explain the possible attacks on the program code given above.

Question 3.

```
#!/bin/sh
grep $1 db.txt > 1.txt
grep $2 db.txt > 2.txt
diff 1.txt 2.txt
```

The above Bash script file named "search" is owned by “ali” user. “ali” has set the setuid bit of this file, and granted read and execute permissions to all users. The script enables to search “db.txt” which is owned by “ali” and finds differences of two search operations on “db.txt”.

Using this code, an attacker wants to read the “/home/ali/secret.txt” file belonging to "ali". How can the attacker achieve this? Explain.

Question 4.

```
<php
if ($username == $allowed && $password ==$secret)
     $authorized = “yes”;
...
if ($authorized == “yes”) {
...
}?>
```

Assume that a web page has the above PHP code. User control is performed with the "username" and "password" variables that come to the page as user parameters, such that “`login.php?username=xxx&password=yyy`”. If the username and password are correct, some priority actions are allowed. How could this page be attacked? Explain.

8 points

Question 5.

```
<HTML>
<head><title>Cell World</title></head>
<body>
<form action="submit_order.php" method="POST">
    <p> Total price for 100 minutes credit is 5.0
    TL. Are you sure you want to order? </p>

    <input type="hidden" name="phone"
    value="05551234567">
    <input type="hidden" name="minute" value="100">
    <input type="hidden" name="price" value="5.0">
    <input type="submit" name="pay" value="Evet">
    <input type="submit" name="pay" value="Hayır">
</form>
</body></HTML>
```

The above HTML code is from a mobile operator's web site. This page is for confirmation of an order to load minutes to a phone number. When the "Yes" button is clicked from this page "submit_order.php" page will be called. Then, the order will be saved to the database and the minute credits (shown with "minute" value) will be loaded to the customer phone (shown with "phone" value). Also, the amount sent with the "price" value is deducted from the customer's credit card. How could this page be attacked?

Your answer

Explain what can be done to prevent this attack by giving examples. 7 points

Your answer

Question 6.

```
$query = "SELECT uid,city FROM users
WHERE nationality='Turkish' AND city='$city';"
```

```
CREATE TABLE users (
  city varchar(20),
  nationality varchar(20),
  uid VARCHAR(20) NOT NULL,
  pass VARCHAR(20) NOT NULL,
  PRIMARY KEY (uid)
);
```

A PHP page's URL is called like this "`query.php?city=Ankara`". The "city" parameter sent to this PHP page is used in the SQL query shown in the top box. Then, the query is executed in the PHP page. Definition of USERS table is given in the second box. How does an attacker perform **below attacks** on this web page? Explain attack details.

List all uid and pass information in USERS table. 8 points

Your answer

Change "pass" field of all users in USERS table. 7 points

Your answer

10 points

Question 7.

```
#include <string.h>
#include <stdio.h>

void foo(const char* input)
{
    char buf[10];
    strcpy(buf, input);
}

void bar(void){
    printf("Augh! I've been hacked!\n");
}

int main(int argc, char* argv[]{
   foo(argv[1]);
   return 0;
}
```

How could an attacker running the above program execute the bar () function using a bug in the program? Explain the attack briefly.

Your answer

Question 8.

Explain how the second order SQL injection attacks can be performed on a web application.

Your answer