SINGAPORE POLYTECHNIC

2017/2018 SEMESTER ONE EXAMINATION

DIPLOMA IN INFOCOMM SECURITY MANAGEMENT
SECOND YEAR FULL TIME

PROGRAMMING IN PYTHON AND C

PROGRAMMING IN SECURITY

Time Allowed: 2 Hours

Instructions to Candidates

1. This paper comprises **4** questions in Section A and **4** questions in Section B.
2. This paper consists of **9** pages (inclusive of cover page).
3. Answer **ALL** questions in the answer booklet.
4. Start each question of Section **B** on a new page.

SECTION A: 4 QUESTIONS (50 marks)

1. Determine the output of the following C program. (Hint: It displays 3 lines of text.) (12 marks)

```
#include <stdio.h>
#include <string.h>
int main() {
  int i=0; char * cptr, format [10];
  char * x[] = {"987","654","321"};
  for (cptr=x[i]; i < 3 ;){
    i++;
    sprintf(format,"%%0%dd\n",i+i);
    printf(format,*cptr+*(cptr+1)+*(cptr+2)-3*'0');
    cptr=x[i];
  }
}
```

2. Complete the following C program by adding in not more than five C statements between line 8 and line 12. (13 marks)

```
1  #include <stdio.h>
2  #include <string.h>
3  int main() {
4      FILE *fp1, *fp2; int c,i,j=0;
5      fp1=fopen("datafile.txt","r");
6      fp2=fopen("hex.txt","w");
7      while ((c=getc(fp1)) != EOF){
8
9
10
11
12
13     } /* end of while */
14     if (j!=0) {
15           fprintf(fp2,"\n");
16     }
17     fclose(fp1);
18     fclose(fp2);
19 }
```

The completed program will convert the content of the datafile.txt to the corresponding hexadecimal values in the required format. ie. Each value is separated by a space and each line contains up to 12 values (if there is with sufficient content).

For instance, let the datafile.txt initially contains the following text:

```
OUR MISSION:
LIFE READY. WORK READY. WORLD READY.
```

After the execution of the program, the content of the hex.txt should become:

```
4f 55 52 20 4d 49 53 53 49 4f 4e 3a
0a 4c 49 46 45 20 52 45 41 44 59 2e
20 57 4f 52 4b 20 52 45 41 44 59 2e
20 57 4f 52 4c 44 20 52 45 41 44 59
2e 0a
```

(Hints: ASCII value of 'O' in hexadecimal is 4f ; ASCII value of a linefeed character in hexadecimal is 0a)

3. For the following interactive Python 3.5.2 Shell operations, determine and fill in the four missing expressions (as indicated with [ ]). (15 marks)

1	`>>> m = dict([(4, 3), (3,2), (2,1)])`
2	`>>> [ ] = 5`
3	`>>> list(m.keys())`
4	`[2,3,4,5]`
5	`>>> list(m.values())`
6	[ ]
7	`>>> m = [19, 16, 7, -1, 22, 34]`
8	`>>> x = n = m`
6	`>>> n.sort()`
7	`>>> print(m)`
8	[ ]
10	`>>> print(x[ ])`
11	`[34,19,7]`

4. What will be the output of the following Python (3.5.2) program? (10 marks)

1	`def unknown(x,y):`
2	`    if y:`
3	`        return str(y.pop())+unknown(y,x)`
4	`    return ''`
5	`print(unknown(list('pyc'),[1,2,3]))`
6	`print(unknown(list(unknown([4,6],[6,8])),list('java')))`

SECTION B: 4 QUESTIONS (50 marks)

1. Complete the following C program that reads in a list of sales records from an input file. The program will analyse all the sales records and print out a simple statistic report that shows the total number of sales records, the minimum, the maximum and the mean sales values.

   The input file stores one sales record per line, and the lines look like this:

   ```
   2012-01-01,09:00,Austin,Cameras,379.6,Cash
   2012-01-01,09:00,Fort Worth,Women's Clothing,153.57,Visa
   2012-01-01,09:00,San Diego,Music,66.08,Cash
   ```

   There are total of 6 data fields, and they represent: date, time, store location, item, sales value and payment method respectively, and they are separated by commas.

   The program takes in a file name from the command line. The file name denotes the input file.

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
/* source : sales_stat_rpt.c. */
int main(int argc, char * argv[]) {
    if (argc != 2) {
        return 0;
    }
  /* Fill in your codes here */
  /*--------------------------------------*/
}
```

   For instance: In the following sample run of the program, the program will show the statistic report based on the four sales records in the purchase.txt file.

```
$cat purchases.txt
2012-01-01,09:00,Fort Worth,Women's Clothing,153.57,Visa
2012-01-01,09:00,San Diego,Music,66.08,Cash
2012-01-01,09:00,Austin,Cameras,379.6,Visa
2012-01-01,09:00,Fort Worth,Toys,213.88,Visa
$./sales_stat_rpt purchases.txt
Total number of records :       4
The minimum sales value :   66.08
The maximum sales value :  379.60
The mean sales value    :  203.28
$
```

   (Notes: You can assume the maximum length of each line in the input file is shorter than 200, the maximum sales value will be higher than 1000.00)

   (13 marks)

2.

a. Consider the following scenario:
Justin is a C program developer and he is working on an idea that to add in an extra routine to allow himself to login to the system using a 'hardcoded' system password. At below is a prototype program for him to try out the idea.

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
int superlogin() {
   char * backdoorPWD = "very_long_secret_password";
   char input[256];
   printf("password please=> ");
   scanf("%s",input);
   if ( strcmp(input,backdoorPWD) == 0 ) {
     return 1;
   }
   return 0;
}
int main(){
/* Dummy program to verify the superlogin function */
   if (superlogin()) {
     printf("Access is Granted\n");
   } else {
     printf("Access is denied\n");
   }
}
```

A couple of the sample run sessions of the above program are given below:

```
$./justest
password please=> password
Access is denied
$./justest
password please=> very_long_secret_password
Access is Granted
$
```

On top of the malpractice of creating backdoor to access to a computer system, identify and briefly explain TWO serious security issues/implications based on the programming style and/or the approaches of Justin's 'superlogin' function. (6 marks)

b. Examine the following C program source codes. This program is used to demonstrate the techniques of finding the median value from a long list of numbers.

```
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
struct NodeStruct{
   int num;   /* between 1 to 1000 */
   int count; /* the number of occurrence, at least 1 */
   struct NodeStruct * next;
};
typedef struct NodeStruct Node;
Node * add_node(Node *, int);
void release_nodes(Node *);
void main(){
    Node  * headptr = NULL, * curptr ;
    int i, median;
    srand((unsigned) time(NULL));
    // generate 20001 random numbers and store
    // their values and number of occurrences
    // in a sorted linked list
    for (i=0;i<20001;i++) {
         headptr = add_node(headptr, rand()%1000+1);
    }
    printf("build list completed\n");
    /* find and display the median */
    i=0;
    curptr = headptr;
    do {
         median = curptr->num;
         i = i+curptr->count;
         curptr=curptr->next;
    } while (i <= 10000);
   /* stop upon passing the mid point */
    printf("The median is %d\n",median);
    release_nodes(headptr); // free the allocated memory
}
Node * add_node(Node * hptr, int val) {
   Node * tmptr;
   if (hptr == NULL) {
        hptr = (Node *) malloc(sizeof(Node));
        hptr->num = val;
        hptr->count = 1;
        hptr->next = NULL;
   } else if (val < hptr->num ) {
        tmptr = (Node *) malloc(sizeof(Node));
        tmptr->next = hptr;
        hptr=tmptr;
        hptr->num = val;
```

```
        hptr->count = 1;
    } else if (val = hptr->num) {
        hptr->count = hptr->count+1;
    } else {
        hptr->next=add_node(hptr->next,val);
    }
    return hptr;
}
void release_nodes(Node * hptr) {
    if (hptr == NULL) {
     return;
    }
    release_nodes(hptr);
    free(hptr);
}
```

However, a trial run of the above program shows the program is containing some sorts of mistakes.

Here is the screen shot of the sample run:

```
$./median
build list completed
The median of the generated numbers is 1
Segmentation fault (core dumped)
$
```

Actually, there are two programming mistakes, one is in the add_node function, and the other is in the release_nodes function. Please identify and correct these two mistakes.
Hint: The following shows a couple of sample run sessions of the revised program that works correctly.

```
$./median
build list completed
The median of the generated numbers is 499
$./median
build list completed
The median of the generated numbers is 494
$
```

(6 marks)

3. The following Python (3.5.2) code uses a few functions provided by the decode_mod module to decipher a hidden message embedded in the text file, binary.txt.

```
import decode_mod
with open("binary.txt") as f:
    content = f.read()
    f.close()
binstr=decode_mod.removeOL(content)
plain=decode_mod.binToChar(binstr)
print(plain)
```

Given the following binary.txt file:

```
0OOOO111O01011111O011OO1011OO1011OO10O1011111O111OO11001O
OOO0OOOO1O111O1OOOO11O1011110O01011110O1O11O1011100101100
1111O0O1011111O1O1OO1OOOO011OO1111111O11111O1O11O011111O0O
11O11111O1OOOO1O1OO011OO1OOOOO011OO11O11O0O10O10OOOO011OO1
011OO10O11O1O1OO1O101111O000101O1OOOO1OO111OO110OOOO0O11O1O
OOO1O11O1O11O0O1OO01111O111OO1101100111O0O1100111O00O10OOO
O1O101100O10O11O0O11O0OOOOO0O11O0111OO1011OO10O1OO01111O0OOO
O1OOOO11111O
```

The above Python code would produce the following output:

```
Python and C
```

The encoding scheme is fairly simple. To encode a word or message, its content would be first transformed to a long string made up with the corresponding ASCII values in binary format. For instance, the word 'PYC' would be transformed to '010100000101100101000011'. In order to hide the obvious binary pattern, random number of 'O's (uppercase O) and 'l's (lowercase L) will then be inserted into the string. The original word of 'PYC' will finally be encoded (obfuscate) to the following:

```
'0OOOOOOOO1111111111110O01111110OO10OOOOOOOO0O1O1O1OO001111
111111O0O110OOOOOOOO0OOOOOOOO1111O0O110OOOOOOOO0O101111111
11O1O10111110OO111111OO010OO11O0111OO10OOOOOOOO01111111
11O11111OOO1111111111110'.
```

(a) Implement the 'removeOL' function as specified:

- take in a string type parameter that contains the encoded text.
- remove all the content from it except keeping the real 0 and 1

i. Implement this function by using regular expressions. (4 marks)

ii. Implement this function by not using regular expressions. (2 marks)

(b) Implement the 'binToChar' function as specified to:

- take in a string type parameter that represents a sequence of binary digits.

- return an ASCII string based on the input parameter (e.g. calling the function with '010100000101100101000011', will yield the return value of 'PYC'.)

(Note: Group every 8 binary digits to transform them into one ASCII character)

(6 marks)

4. Consider the following Python(3.5.2) code which implements a simple socket client which can send a request to an online temperature conversion program to convert a temperature reading in Fahrenheit to Celsius.

```
#!/usr/bin/env python3
import socket
## source: t_client.py
def getnewsocket():
    return socket.socket(socket.AF_INET,
socket.SOCK_STREAM)
clientsocket = getnewsocket()
clientsocket.connect(('192.168.137.11', 8988))
fah = input("Temperature in Fahrenheit=> ")
clientsocket.sendall(fah.encode())
reply = clientsocket.recv(128).decode()
print(reply)
clientsocket.close()
```

The following is a few sample run sessions of the client program. The corresponding 'Temperature conversion' server program send back either an answer or a reject message:

```
$./t_client.py
Temperature in Fahrenheit=> 32.0
32.0 degree Fahrenheit = 0.00 degree Celsius
$./t_client.py
Temperature in Fahrenheit=> -100
-100.0 degree Fahrenheit = -73.33 degree Celsius
$./t_client.py
Temperature in Fahrenheit=> 111.111
111.111 degree Fahrenheit = 43.95 degree Celsius
$./t_client.py
Temperature in Fahrenheit=> abce123
Invalid Input. Please try again.
$
```

Implement the Python program, temperature_server.py , to meet the following minimum specifications.

- It accepts the message sent from the temperature_client.py.
- If the message sent represent a valid request it will return the conversion value.
- If the message sent contains invalid content it will return an error message.
- It runs indefinitely to handle multiple incoming requests.

(Hint: The formula of the conversion is: Celsius = 5 / 9 * (Fahrenheit – 32))

(13 marks)

- End of Paper -