Computer Programming

Practical File

Kushagra Lakhwani
2021UCI8036

Contents

1 The C Language

1.1 Student Grades

1.1.1 Statement

Write a program in C that uses a two-dimensional array to store the numeric grade for each student (n) in a multiple teacher's class (m). The program assumes that the teacher has three classes and a maximum of 30 students per class. Both the variable M and N should be user defined.

1.1.2 Code

```c
#include <stdio.h>

struct student
{
  char grade;
};

void get_marks(student classes[][30])
{
  int m, n;

  printf("\nEnter class: ");
  scanf("%d", &m);
  printf("Enter student: ");
  scanf("%d", &n);

  printf("The given student has got %c grade!\n", classes[m - 1][n - 1]);
}

void populate_classes(student classes[3][30])
{
  int num;

  for (int i = 0; i < 3; i++)
  {
    student *curr_class = classes[i];
    printf("Enter number of students in class %d: ", i + 1);
    scanf("%d", &num);

    for (int j = 0; j < num; j++)
    {
      printf("Enter the Grade of student %d in class %d: ", j + 1, i + 1);
      scanf(" %c", &(curr_class[j].grade)); // blank space is important
    }
    printf("\n");
  }
```

```
}

int main(int argc, char const *argv[])
{

  student classes[3][30];

  populate_classes(classes);
  get_marks(classes);

  return 0;
}
```

1.1.3 Output

```
➜ Code ./student_grades
Enter number of students in class 1: 3
Enter the Grade of student 1 in class 1: A
Enter the Grade of student 2 in class 1: B
Enter the Grade of student 3 in class 1: C

Enter number of students in class 2: 2
Enter the Grade of student 1 in class 2: A
Enter the Grade of student 2 in class 2: B

Enter number of students in class 3: 2
Enter the Grade of student 1 in class 3: F
Enter the Grade of student 2 in class 3: A


Enter class: 2
Enter student: 1
The given student has got A grade!
➜ Code
```

Figure 1: Output - 1

1.2 Average Age

1.2.1 Statement

Write the program to input the value of age of employees in the company. You have to calculate the average age of the employee in the company using pointer of array.

1.2.2 Code

```c
#include <stdio.h>
#include <stdlib.h>

float avg(int arr[], int n)
{
  float sum = 0;
  for (int i = 0; i < n; i++)
    sum += arr[i];
  return sum / (float)n;
}

int main(int argc, char const *argv[])
{
  int n;
  printf("Enter number of employees: ");
  scanf("%d", &n);

  int *employees = (int *)malloc(n * sizeof(int));

  for (int i = 0; i < n; i++)
  {
    printf("Enter age of employee %d: ", i + 1);
    scanf("%d", &employees[i]);
  }

  printf("The average age of employees is = %.2f\n", avg(employees, n));

  free(employees);

  return 0;
}
```

1.2.3 Output

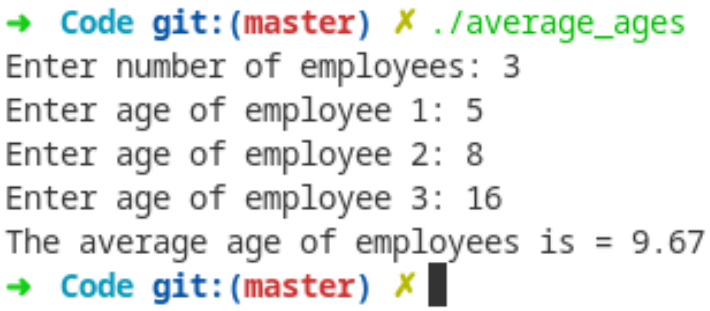

Figure 2: Output - 2

1.3 Length of string

1.3.1 Statement

A user has given a random size string to input, you have to calculate the length of the string using pointer. You cannot use predefined function strrev.

1.3.2 Code

```c
#include <stdio.h>
int main()
{
  char str[100], i;
  printf("Enter a string: ");
  scanf("%[^\n]s", str);

  // '\0' represents end of String
  for (i = 0; str[i] != '\0'; ++i)
    ;
  printf("\nLength of input string: %d\n", i);

  return 0;
}
```

1.3.3 Output

```
➜ Code git:(master) ✗ ./strlen
Enter a string: this is a string that is given to the program

Length of input string: 45
➜ Code git:(master) ✗
```

Figure 3: Output - 3

1.4 Start-Up Owner

1.4.1 Statement

A start-up owner is interested to maintain the dataset of the newly recruited employees.

She is interested in storing the Emp_Name (Str), Emp_Age (int), Emp_Degree (Str), Emp_Exp (Float), Emp_add (Structure). Emp_add needs one user defined data to store street no, city, district and state for the employee address. You have to design a database where we can store all the information for at least 20 employees.

1.4.2 Code

```
#include <stdio.h>

struct address
{
  int street;
  char city[50];
  char district[50];
  char state[50];
};

struct employee
{
  char Emp_Name[50];
  int Emp_Age;
  char Emp_Degree[50];
  float Emp_Exp;
  address Emp_add;
};

void add_emploee(employee &Employee, int i)
{
  printf("\nEnter Name of employee %d: ", i);
  scanf("%s", &Employee.Emp_Name);
  printf("Enter Age of employee %d: ", i);
  scanf("%d", &Employee.Emp_Age);
  printf("Enter Degree of employee %d: ", i);
  scanf("%s", &Employee.Emp_Degree);
  printf("Enter Experience of employee %d: ", i);
  scanf("%f", &Employee.Emp_Exp);

  printf("*Address Details*\n");
  printf("Enter City of employee %d: ", i);
  scanf("%s", &Employee.Emp_add.city);
  printf("Enter District of employee %d: ", i);
  scanf("%s", &Employee.Emp_add.district);
  printf("Enter State of employee %d: ", i);
```

```
  scanf("%s", &Employee.Emp_add.state);
  printf("Enter Street of employee %d: ", i);
  scanf("%d", &Employee.Emp_add.street);
}

void print_employee(employee Employee)
{
  printf("\n%s %s %s", Employee.Emp_Name, Employee.Emp_Degree, Employee.Emp_
}

int main(int argc, char const *argv[])
{
  int n;
  printf("Enter Number of employees: ");
  scanf("%d", &n);

  employee Employees[n];

  for (int i = 0; i < n; i++)
    add_emploee(Employees[i], i + 1);

  for (int i = 0; i < n; i++)
    print_employee(Employees[i]);
  return 0;
}
```

1.4.3 Output

```
➜  Code git:(master) ✗ ./employee
Enter Number of employees: 1

Enter Name of employee 1: John
Enter Age of employee 1: 23
Enter Degree of employee 1: Betch
Enter Experience of employee 1: 2
*Address Details*
Enter City of employee 1: Delhi
Enter District of employee 1: Dwarka
Enter State of employee 1: Delhi
Enter Street of employee 1: 3

John Betch Delhi%
➜  Code git:(master) ✗
```

Figure 4: Output - 4

1.5 Student Names

1.5.1 Statement

Defined a two-dimensional matrix (char)[50][20] to store the student's name in the class. We are expecting to store the 50 students with different length name. Write a program to print all the name with the help of pointers

1.5.2 Code

```c
#include <stdio.h>

void print_names(char students[][20], int n)
{
  printf("\nStudents are: \n");
  for (int i = 0; i < n; i++)
    printf("%s\n", *(students + i));
}

int main(int argc, char const *argv[])
{
  int n;
  char students[50][20];
  printf("Enter a number: ");
  scanf("%d", &n);

  for (int i = 0; i < n; i++)
  {
    printf("Enter the name for student %d: ", i + 1);
    scanf("%s", *(students + i));
  }
  print_names(students, n);
  return 0;
}
```

1.5.3 Output

```
➜ Code git:(master) ✗ ./students
Enter a number: 4
Enter the name for student 1: John
Enter the name for student 2: KorigamiK
Enter the name for student 3: doe
Enter the name for student 4: aka

Students are:
John
KorigamiK
doe
aka
➜ Code git:(master) ✗
```

Figure 5: Output - 5

1.6 XOR Operation

1.6.1 Statement

The outcome of a XOR operation is true if and only if one operand (but not both) is true. Write a program in 'C' which returns the outcome of an Exclusive OR operation performed on its two operands

1.6.2 Code

```c
#include <stdio.h>

int main(int argc, char const *argv[])
{
  int a, b;
  printf("Enter 2 numbers:\n");
  scanf("%d", &a);
  scanf("%d", &b);
  int xor_result = (a | b) & (~(a & b));
  printf("The XOR of %d and %d is equal to %d\n", a, b, xor_result);
  return 0;
}
```

1.6.3 Output

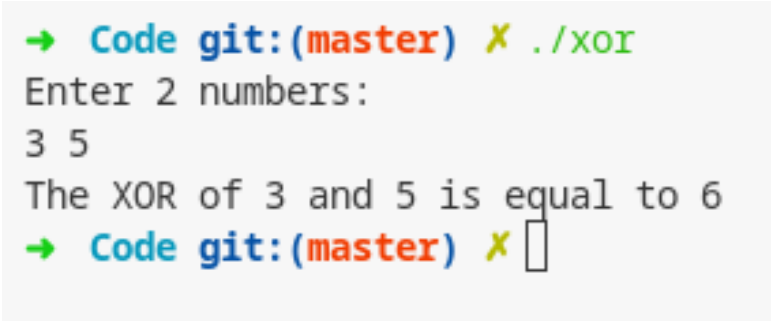

Figure 6: Output - 6

1.7 Left and Right Shift

1.7.1 Statement

Write a program in C to show that Right shift effectively divides a number by 2 and a left shift effectively multiplies a number by 2

1.7.2 Code

```c
#include <stdio.h>

int main(int argc, char const *argv[])
{
  int a;
  printf("Enter a numbers: ");
  scanf("%d", &a);

  for (int i = 0; i < 4; i++)
    printf("%d Right shifted %d times is equal to %d\n", a, i, a >> i);
  printf("\n");
  for (int i = 0; i < 4; i++)
    printf("%d Left shifted %d times is equal to %d\n", a, i, a << i);

  return 0;
}
```

1.7.3 Output

```
➜  Code git:(master) ✗ ./shifts
Enter a numbers: 20
20 Right shifted 0 times is equal to 20
20 Right shifted 1 times is equal to 10
20 Right shifted 2 times is equal to 5
20 Right shifted 3 times is equal to 2

20 Left shifted 0 times is equal to 20
20 Left shifted 1 times is equal to 40
20 Left shifted 2 times is equal to 80
20 Left shifted 3 times is equal to 160
➜  Code git:(master) ✗
```

Figure 7: Output - 7

1.8 Magic Number

1.8.1 Statement

Using the ? Operator, rewrite the magic number program discussed in the class

1.8.2 Code

```c
#include <stdio.h>

bool is_magic(int num)
{
  int sum_of_digits = 0, reversed = 0, original = num;
  while (num)
  {
    sum_of_digits += num % 10;
    reversed = reversed * 10 + (num % 10);
    num /= 10;
  }
  return sum_of_digits * reversed == original;
}

int main(int argc, char const *argv[])
{
  int number;
  printf("Enter a number: ");
  scanf("%d", &number);

  is_magic(number)
      ? printf("It is a Magic Number!\n")
      : printf("Not a Magic number :(\n");

  return 0;
}
```

1.8.3 Output

```
➜ Code git:(master) ✗ ./magic_number
Enter a number: 23
Not a Magic number :(
➜ Code git:(master) ✗
```

Figure 8: Output - 8

1.9 Quotient

1.9.1 Statement

Using if else statement write a program in 'C' to read two integers from the user and display the quotient. Your program should be able to detect divide by zero.

1.9.2 Code

```c
#include <stdio.h>

int main(int argc, char const *argv[])
{
  float a, b;
  printf("Enter 2 numbers:\n");
  scanf("%f", &a);
  scanf("%f", &b);
  b == 0
      ? printf("Undefined Behavior\n")
      : printf("The quotient is: %.2f\n", a / b);
  return 0;
}
```

1.9.3 Output

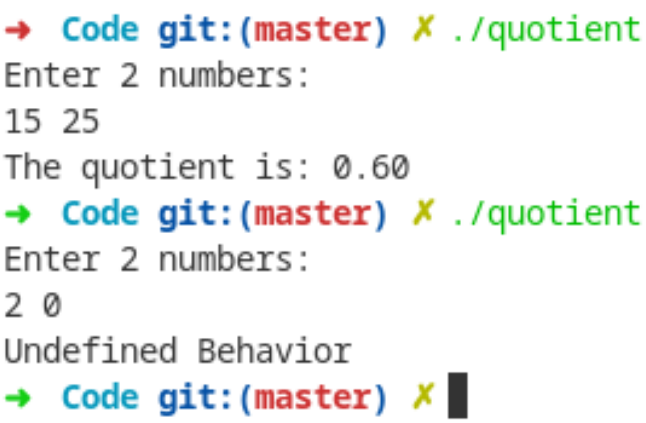

Figure 9: Output - 9

1.10 Text File

1.10.1 Statement

Write a program in C that inputs lines of text until a blank line is entered. Then it redisplays each line one character at a time

1.10.2 Code

```c
#include <stdio.h>

int main(int argc, char const *argv[])
{
  char x, text[100];
  int i = 0;
  while (x = getchar())
  {
    if (x == '\n')
    {
      for (int j = 0; j < i; j++)
        printf("%c\n", text[j]);
      break;
    }
    else
      text[i] = x;
    i++;
  }

  return 0;
}
```

1.10.3 Output

Figure 10: Output - 10

1.11 Queue

1.11.1 Statement

Write a program in C using pointers to implement insertion and deletion in a queue. A queue is a data structure that follows a first in first out i.e. the element to go in first is the one to come out first

1.11.2 Code

```c
#include <stdio.h>
#define SIZE 5

void enQueue(int);
void deQueue();
void display();

int items[SIZE], front = -1, rear = -1;

void enQueue(int value)
{
  if (rear == SIZE - 1)
    printf("\nQueue is Full!!");
  else
  {
    if (front == -1)
      front = 0;
    rear++;
    items[rear] = value;
    printf("\nInserted -> %d", value);
  }
}

void deQueue()
{
  if (front == -1)
    printf("\nQueue is Empty!!");
  else
  {
    printf("\nDeleted : %d", items[front]);
    front++;
    if (front > rear)
      front = rear = -1;
  }
}

// Function to print the queue
void display()
{
```

```
  if (rear == -1)
    printf("\nQueue is Empty!!!");
  else
  {
    int i;
    printf("\nQueue elements are:\n");
    for (i = front; i <= rear; i++)
      printf("%d  ", items[i]);
  }
  printf("\n");
}

int main()
{
  // deQueue is not possible on empty queue
  deQueue();

  // enQueue 5 elements
  enQueue(1);
  enQueue(2);
  enQueue(3);
  enQueue(4);
  enQueue(5);

  // 6th element can't be added to because the queue is full
  enQueue(6);

  display();

  // deQueue removes element entered first i.e. 1
  deQueue();

  display();

  return 0;
}
```

1.11.3 Output

```
➜  Code git:(master) ✗ ./q

Queue is Empty!!
Inserted -> 1
Inserted -> 2
Inserted -> 3
Inserted -> 4
Inserted -> 5
Queue is Full!!
Queue elements are:
1  2  3  4  5

Deleted : 1
Queue elements are:
2  3  4  5
➜  Code git:(master) ✗
```

Figure 11: Output - 11

1.12 Temperature Conversion

1.12.1 Statement

Write a program to print the corresponding celsius to Fahrenheit table. Modify the temperature conversion program to print the table in reverse order, that is from 300 to 0

1.12.2 Code

```c
#include <stdio.h>

double to_celcius(int fahr)
{
  return (fahr - 32) * (5.0 / 9.0);
}

int main()
{
  int fahr;

  printf("Fahrenheit      \t Celcius\n");
  for (fahr = 300; fahr >= 0; fahr = fahr - 20)
    printf("%3d \t\t %6.1f\n", fahr, to_celcius(fahr));
}
```

1.12.3 Output

```
➜ Code git:(master) ✗ g++ ./temperature.cpp -o temperature && ./temperature
Fahrenheit       Celcius
300               148.9
280               137.8
260               126.7
240               115.6
220               104.4
200                93.3
180                82.2
160                71.1
140                60.0
120                48.9
100                37.8
 80                26.7
 60                15.6
 40                 4.4
 20                -6.7
  0               -17.8
➜ Code git:(master) ✗
```

Figure 12: Output - 12

1.13 Text in File

1.13.1 Statement

Write a program to count blanks, tabs and newlines

1.13.2 Code

```c
#include <stdio.h>

int main()
{
  int blank_char = 0, tab_char = 0, new_line = 0, c;
  printf("Number of blanks, tabs, and newlines:\n");
  printf("Input few words/tab/newlines\n");
  while ((c = getchar()) != EOF)
  {
    switch (c)
    {
    case ' ':
      ++blank_char;
      break;
    case '\t':
      ++tab_char;
      break;
    case '\n':
      ++new_line;
      break;
    default:
      break;
    }
  }
  printf("\nblank=%d, tab=%d, newline=%d\n", blank_char, tab_char, new_line
}
```

1.13.3 Output

```
➜ Code git:(master) ✗ ./input_info
Number of blanks, tabs, and newlines:
Input few words/tab/newlines
 laks dflkja sdjfoaisjd ofajs dflkja


blank=5, tab=0, newline=2
➜ Code git:(master) ✗
```

Figure 13: Output - 13

1.14 Frequencies

1.14.1 Statement

Write a program to print the histogram of the frequencies of different characters of its input.

1.14.2 Code

```c
#include <stdio.h>
#include <string.h>
void histogram(const int offset, const int range)
{
  FILE *file = fopen("./text.txt", "r+");
  int histogram[range];
  memset(histogram, 0, sizeof(histogram)); // initialize 95 spaces for ASCI

  int special = 0;

  int c;
  while ((c = fgetc(file)) != EOF)
  {
    if (c < offset || c >= (offset + range))
      special++;
    else
      ++histogram[c - offset];
  }

  for (int i = 0; i < range; ++i)
  {
    c = i + offset;
    printf("%c ", c);
    for (int j = 0; j < histogram[i]; ++j)
      putchar('x');
    putchar('\n');
  }

  printf("- ");
  for (int j = 0; j < special; j++)
    putchar('x');
  putchar('\n');
}

int main(void)
{
  histogram(' ', 95); // ' ' is 32 in ascii
}
```

1.14.3 Output

Input Text

```
➜  Code git:(master) ✗ cat ./text.txt
Lorem ipsum
From Wikipedia, the free encyclopedia
Jump to navigation
Jump to search
"Ipsum" redirects here. For the car, see Toyota Ipsum.
An example of the Lorem ipsum placeholder text on a green and white webpage.
Using Lorem ipsum to focus attention on graphic elements in a webpage design proposal

In publishing and graphic design, Lorem ipsum is a placeholder text commonly used to demonstrate the visual form of a
 document or a typeface without relying on meaningful content. Lorem ipsum may be used as a placeholder before the fi
nal copy is available. It is also used to temporarily replace text in a process called greeking, which allows designe
rs to consider the form of a webpage or publication, without the meaning of the text influencing the design.

Lorem ipsum is typically a corrupted version of De finibus bonorum et malorum, a 1st-century BC text by the Roman sta
tesman and philosopher Cicero, with words altered, added, and removed to make it nonsensical and improper Latin.
➜  Code git:(master) ✗
```

Figure 14: Output - 14 - Text

Histogram

```
➜ Code git:(master) ✗ ./histogram
  xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
!
" xx
#
$
%
&
'
(
)
*
+
, xxxxxxxxx
- x
. xxxxxxx
/
0
1 x
2
3
4
5
6
7
8
9
:
;
<
=
>
?
@
A x
B x
C xx
D x
E
F xx
G
H
I xxxx
J xx
K
L xxxxxxx
M
N
O
P
Q
R x
S
T x
U x
V
W x
X
Y
Z
[
\
]
^
_
`
a xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
b xxxxxxxxxxx
c xxxxxxxxxxxxxxxxxxxxxxxxxxxx
d xxxxxxxxxxxxxxxxxxxxxxxxxxxxx
e xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
f xxxxxxxxxxxxxxx
g xxxxxxxxxxxxxxxxxxx
h xxxxxxxxxxxxxxxxxxxxxxxxxx
i xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
j
k xxx
l xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
m xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
n xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
o xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
p xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
q
r xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
s xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
t xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
u xxxxxxxxxxxxxxxxxxxxxxxxxxx
v xxxxx
w xxxxxxxxxx
x xxxxxx
y xxxxxxxxxxxx
z
{
|
}
~
- xxxxxxxxxx
➜ Code git:(master) ✗
```

Figure 15: Output - 14 - Histogram

2 The Python Programming Language

2.1 Linear Search

2.1.1 Code

```
flag = 0
x = []
for i in range(0, 5, 1):
    x.append(int(input("Enter No. ")))

print(*x, sep=' ')

y = int(input("Enter any No. "))

for i in range(0, 5, 1):
    if x[i] == y:
        flag = 1
        break
if flag == 1:
    print("Found")
else:
    print("Not Found")
```

2.1.2 Output

Figure 16: Output - py - 1

2.2 Binary Search

2.2.1 Code

```
l = []
n = int(input("Enter no. of Elements: "))
for i in range(0, n, 1):
    a = int(input("Enter any no. "))
    l.append(a)
l.sort()
print(*l, sep=' ')

x = int(input("Enter the Value to be Searched: "))
flag = 0
beg = 0
back = n - 1
while beg <= back:
    mid = (beg + back) // 2
    if l[mid] == x:
        flag = 1
        break
    elif x > l[mid]:
        beg = mid + 1
    else:
        back = mid - 1
if flag == 1:
    print("Found at", mid, "index")
else:
    print("Not Found")
```

2.2.2 Output

Figure 17: Output - py - 2

2.3 Bubble Sort

2.3.1 Code

```
n = int(input("Enter No. of Digits: "))
x = []
for i in range(0, n, 1):
    x.append(int(input("Enter No. ")))
print(*x, sep=' ')

for i in range(0, n - 1, 1):
    for j in range(0, n - 1, 1):
        if x[j] > x[j + 1]:
            x[j], x[j + 1] = x[j + 1], x[j]
print(*x, sep=' ')  # sorted array
```

2.3.2 Output

Figure 18: Output - py - 3

2.4 Selection Sort

2.4.1 Code

```
x = []
n = int(input("Enter no. of Elements: "))
for i in range(0, n, 1):
    a = int(input("Enter any No. "))
    x.append(a)

for i in range(0, n - 1, 1):
    pos = i
    em = x[i]
    for j in range(i + 1, n, 1):
        if x[j] < em:
            em = x[j]
            pos = j
        t = x[i]
        x[i] = x[pos]
        x[pos] = t

print(*x, sep=' ')
```

2.4.2 Output

Figure 19: Output - py - 4

2.5 Insertion Sort

2.5.1 Code

```
def insertionSort(arr):
    for i in range(1, len(arr)):
        key = arr[i]
        j = i - 1
        while j >= 0 and key < arr[j]:
            arr[j + 1] = arr[j]
            j -= 1
        arr[j + 1] = key


arr = [12, 11, 13, 5, 6]
insertionSort(arr)

print("Sorted array is:")
print(*arr, sep=' ')
```

2.5.2 Output

```
➜  Code git:(master) ✗ /bin/python3 -u
Sorted array is:
5 6 11 12 13
➜  Code git:(master) ✗
```

Figure 20: Output - py - 5

2.6 Copy File

2.6.1 Statement

Write a Python program to copy from an existing file named “text.txt” to another file “text_copy.txt”.

2.6.2 Code

```
#! /usr/bin/python

with open('text.txt', 'r+') as in_file:
    with open('text_copy.txt', 'w+') as out_file:
        for line in in_file:
            out_file.write(line)
```

2.6.3 Output

```
➜  Code git:(master) ✗ ./file_copy.py
➜  Code git:(master) ✗ cat ./text_copy.txt
Lorem ipsum
From Wikipedia, the free encyclopedia
Jump to navigation
Jump to search
"Ipsum" redirects here. For the car, see Toyota Ipsum.
An example of the Lorem ipsum placeholder text on a green and white webpage.
Using Lorem ipsum to focus attention on graphic elements in a webpage design proposal

In publishing and graphic design, Lorem ipsum is a placeholder text commonly used to demonstr
 document or a typeface without relying on meaningful content. Lorem ipsum may be used as a p
```

Figure 21: Output - py - 6

List of Figures