Vidya Jyothi Institute of Technology

(An Autonomous Institution)

(Accredited by NAAC, Approved by AICTE & Permanently Affiliated to JNTUH)

DEPARTMENT OF INFORMATION TECHNOLOGY

Course Name : Data Structures & Python programming Lab

Course ID : A43583

Prerequisites : C Programming,Mathematics and Basics of computer

Name of the Faculty: Dr.M.Nagabhushana Rao/G.Indira Priyadarshini

B.Tech-Semseter

(R20)

Program Outcomes-POs :

1. **Engineering Knowledge**: Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.
2. **Problem analysis**: Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.
3. **Design/development of solutions**: Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.
4. **Conduct investigations of complex problems**: Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.
5. **Modern tool usage**: Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations.
6. **The engineer and society**: Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.
7. **Environment and sustainability**: Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.
8. **Ethics**: Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.
9. **Individual and team work**: Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.
10. **Communication**: Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.
11. **Project management and finance**: Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.
12. **Life-long learning**: Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

Program Specific Outcomes:

Program Specific Outcomes	Statements
PSO1	Enhanced ability in applying mathematical abstraction and algorithmic design along with programming tools to solve complexity involved in efficient programming.
PSO2	Develop effective software skills and documentation ability for graduates to become Employable/ Higher studies/Entrepreneur researcher.

Course Outcomes(COs):

Course name: DATA STRUCTURES & PYTHON PROGRAMMING LAB

After completing this course the student must demonstrate the knowledge and ability to	
CO1	Develop the programs on stacks, trees and its applications.
CO2	Design and implementation of programs on BST and Graph Traversals.
CO3	Apply Hashing techniques in real world applications
CO4	Implement oops concepts in Python
CO5	Develop Programs on modules and Packages
CO6	Design Programs that handle errors

CO-PO Mappings:

Course name: Data Structures and Python Programming Lab

	PO 1	PO 2	PO 3	PO 4	PO 5	PO 6	PO 7	PO 8	PO 9	PO 10	PO 11	PO 12
CO 1	3	3	3	3	3	2			2	2	1	3
CO 2	3	3	3	3	3	2			2	2	1	3
CO 3	3	3	3	3	3	2			2	2	1	3
CO 4	3	3	3	3	3	2			2	2	1	3
CO 5	3	3	3	3	3	2			2	2	1	3
Avg	3	3	3	3	3	2			2	2	1	3

CO-PSO Mappings:

Course name: Data Structures and Python Programming Lab

	PSO1	PSO2
CO1	3	3
CO2	3	3
CO3	3	3
CO4	3	3
CO5	3	3
AVG	3	3

S. No.	Name of the Experiment	Page no
1	Write a program to illustrate concepts of arrays, structures, unions and enumerated data types.	2 - 3
2	Write a program to convert infix to post fix notation	4 - 7
3	Write a program to evaluate postfix notations	8 - 10
4	Write a program to illustrate tree traversals a)In order b)Preorder c)Post order	11 - 13
5	Write a program to illustrate insertion, deletion and searching in Binary Search Tree.	14 - 18
6	Write a program to illustrate Insertion, deletion and Rotation on AVL Tree.	19 - 24
7	Write a program to illustrate Graph traversals a) Breadth First Search b) Depth First Search	25 - 28
8	Write a program to implement hash table using linear and quadratic probing	29 - 38

Python Programming Lab

Syllabus

Exercise 1

a) Installation and Environment setup of python.

b) Write a program to demonstrate the use of basic Data Types

c) Write a program to demonstrate the Operators and Expressions

d) Write a program to demonstrate the Functions and parameter passing Techniques.

Exercise 2

a) Write a Program to implement

i. Packages **ii**. Modules **iii**. Built-in Functions

b) Write a Program to implement

i. List **ii**. Tuple **iii**. Dictionaries

c) Programs on Stings, String Operations and Regular Expressions

Exercise 3

a) Write a Program to implement Class and Object

b) Write a Program to implement Static and Instance methods, Abstract Classes and Interfaces.

Exercise 4

a) Write a program to compute distance between two points taking input from the user (Pythagorean Theorem)

b) Write a program to convert a given decimal number to other base systems

Exercise 5

a) Write a program to implement Inheritance

b) Write a program to implement Polymorphism

Exercise 6

a) Write a program to implement Files

b) Write a program to Exception Handling**.**

Experiment - 1

Write a program to illustrate concepts of arrays, structures, unions and enumerated data types.

Arrays:

```
#include <stdio.h>

int main() {
  int values[5];

  printf("Enter 5 integers: ");

  // taking input and storing it in an array
  for(int i = 0; i < 5; ++i) {
    scanf("%d", &values[i]);
  }

  printf("Displaying integers: ");

  // printing elements of an array
  for(int i = 0; i < 5; ++i) {
    printf("%d\n", values[i]);
  }
  return 0;
}
```

Output

```
Enter 5 integers: 1
-3    34     0    3
Displaying integers: 1
-3
34
0
3
```

Structures:

```
struct Point
{
  int x, y;
};

int main()
{
  struct Point p1 = {0, 1};
```

```
  // Accessing members of point p1
  p1.x = 20;
  printf ("x = %d, y = %d", p1.x, p1.y);

  return 0;
}
```

Output:

x = 20, y = 1

Unions:

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

union Data {
  int i;
  float f;
  char str[20];
};

int main( ) {

  union Data data;

  printf( "Memory size occupied by data : %d\n", sizeof(data));

  return 0;
}
```

Output:

Memory size occupied by data : 20

Enum:

```
#include<stdio.h>
enum week{Mon=10, Tue, Wed, Thur, Fri=10, Sat=16, Sun};
enum day{Mond, Tues, Wedn, Thurs, Frid=18, Satu=11, Sund};
int main() {
  printf("The value of enum week: %d\t%d\t%d\t%d\t%d\t%d\t%d\n\n",Mon , Tue, Wed, Thur,
Fri, Sat, Sun);
  printf("The default value of enum day: %d\t%d\t%d\t%d\t%d\t%d\t%d",Mond , Tues, Wedn,
Thurs, Frid, Satu, Sund);
  return 0;
}
```

Output

```
The value of enum week: 1011    12    13    10    16    17
The default value of enum day: 0    1    2    3    18    11    12
```

Experiment 2

Aim: Write a C program that uses stack operations to convert a given infix expression into its postfix Equivalent, Implement the stack using an array.

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

char stack[20];
int top=-1;
char pop();  /*declaration of  pop function*/
void push(char item); /*declaration of  push function*/
intprcd(char symbol)  /*checking the precedence*/
{
switch(symbol) /*assigning values for symbols*/
  {
case '+':
case '-': return 2;
                    break;

          case '*':
          case '/':
          case '%': return 4;
                    break;
          case '(':
          case ')':
          case '#': return 1;
                    break;

 }

}
intisoperator(char symbol)  /*assigning  operators*/
{
switch(symbol)
  {
```

```
case '+':
case '*':
case '-':
case '/':
case '%':
case '(':
case ')':return 1;
        break;
default: return 0;
  }
}

/*converting infix to postfix*/
voidconvertip(char infix[],char postfix[])
{
        inti,n,j=0;
char symbol;
        top++;
        stack[top]='#';
        n=strlen(infix);
        for(i=0;i<strlen(infix);i++)
        {
        symbol=infix[i];
                if(isoperator(symbol)==0)
                {
                        postfix[j]=symbol;
                        j++;
                 }
                else
                 {
                        if(symbol=='(')
                        push(symbol); /*function call  for pushing elements into the stack*/
                        else if(symbol==')')
                        {
                                while(stack[top]!='(')
                                {
                                postfix[j]=pop();
                                j++;
```

```
                        }
                        pop();  /*function call  for popping elements into the stack*/
                    }
                    else
                    {
                    if(prcd(symbol)>prcd(stack[top]))
                    push(symbol);
                    else
                     {
                                while(prcd(symbol)<=prcd(stack[top]))
                                {
                                postfix[j]=pop();
                                j++;
                                }
                                push(symbol);

                      }/*end of else */
                    }/*end of else */
            } /*end of else */
        }/*end of for loop*/

        while(stack[top]!='#')
        {
                postfix[j]=pop();
                j++;
        }
        postfix[j]='\0'; /*null terminate string*/
}

/*main program*/
void main()
 {

char infix[20],postfix[20];
//clrscr();
        printf("enter the valid infix string \n");
        gets(infix);
convertip(infix,postfix);   /*function call   for converting infix to postfix */
```

```
printf("the corresponding postfix string is:\n");
puts(postfix);
getch();
  }

/*push operation*/
void push(char item)
{
top++;
          stack[top]=item;
 }
/*pop operation*/
char pop()
{
char a;
   a=stack[top];
top--;
return a;
 }
```

Output:

```
enter the valid infix string
9-((3*4)+8/4
the corresponding postfix string is: 934*8+-4/
```

Experiment 3

Aim: Write a C program to evaluate postfix notations

```
#include<stdio.h>
#include<conio.h>
#include<string.h>
#define MAX 50
int stack[MAX];
char postfix[20];
int top=-1;
void push(int item);
char pop();
void evaluate(char s);
void main()
{
inti,n,x;
char symbol;
// clrscr();
printf("Insert a postfix notation :: \n");
gets(postfix);
 n=strlen(postfix);
for(i=0;i<n;i++)
          {
symbol=postfix[i];
                    if(symbol>='0' && symbol<='9')
                     {
                      x=(int)(symbol-48);
                    push(x);
                    }
                    else
                     {
                    evaluate(postfix[i]);
                     }
 }
printf("\n\nResult is :: %d",stack[top]);
getch();
}
void push(int x)
```

```
{
top++;
stack[top]=x;
}
char pop()
{
        char x;
         x=stack[top];
        top--;
        return x;
}

void evaluate(char d)
{
inta,b,c;
 a=pop();
 b=pop();

        switch(d)
        {
                case '+':
                          c=a+b;
                        top++;
                        stack[top]=c;
                        break;
                case '-':
                        c=a-b;
                        top++;
                        stack[top]=c;
                        break;
                case '*':
                        c=a*b;
                        top++;
                        stack[top]=c;
                        break;
                case '/':
                        c=a/b;
                        top++;
```

```
                    stack[top]=c;
                    break;
        case '%':
                    c=a%b;
                    break;
                    top++;
                    stack[top]=c;
        default:
                     c=0;
                }

}
```

OUTPUT:

Insert a postfix notation: 432+*
Result is: 20

Experiment 4

Aim: C program to illustrate tree traversals
a) In order b) preorder c) post order

In-order traversal method:
1. Visit Left Sub-Tree
2. Process Current Node
3. Visit Right Sub-Tree

Pre-order traversal method:
1. Process Current Node
2. Visit Left Sub-Tree
3. Visit Right Sub-Tree

Post-order traversal method:
1. Visit Left Sub-Tree
2. Visit Right Sub-Tree
3. Process Current Node

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

struct node
{
int value;
node* left;
node* right;
};

struct node* root;

struct node* insert(struct node* r,int data);
voidinOrder(struct node* r);
voidpreOrder(struct node* r);
voidpostOrder(struct node* r);

int main()
{
root= NULL;
int n, v;

printf("How many data's do you want to insert ?\n");
scanf("%d",&n);
```

```
for(int i=0; i<n; i++){
printf("Data %d: ", i+1);
scanf("%d",&v);
root= insert(root, v);
}

printf("Inorder Traversal: ");
inOrder(root);
printf("\n");

printf("Preorder Traversal: ");
preOrder(root);
printf("\n");

printf("Postorder Traversal: ");
postOrder(root);
printf("\n");


return0;
}

struct node* insert(struct node* r,int data)
{
if(r==NULL)
{
      r =(struct node*)malloc(sizeof(struct node));
r->value = data;
r->left = NULL;
r->right = NULL;
}
elseif(data < r->value){
r->left = insert(r->left, data);
}
else{
r->right = insert(r->right, data);
}
return r;

}

void inOrder(struct node* r)
{
```

```
if(r!=NULL){
inOrder(r->left);
printf("%d ", r->value);
inOrder(r->right);
}
}

void preOrder(struct node* r)
{
if(r!=NULL){
printf("%d ", r->value);
preOrder(r->left);
preOrder(r->right);
}
}

void postOrder(struct node* r)
{
if(r!=NULL){
postOrder(r->left);
postOrder(r->right);
printf("%d ", r->value);

}
}
```

OUTPUT:

```
How many data's do you want to insert? 7
20  15  25  16  24   12   30

Inorder Traversal:      12    15  16  20  24  25    30
Preorder Traversal:    20  15  12  16    25    24   30
Postorder Traversal:   12   16   15   24   30  25  20
```

Experiment 5

Aim: C program to illustrate insert, delete and searching operation in binary search tree

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

struct node
{
        int key;
        struct node *left, *right;
};

// A utility function to create a new BST node
struct node *newNode(int item)
{
        struct node *temp = (struct node *)malloc(sizeof(struct node));
        temp->key = item;
        temp->left = NULL;
        temp->right = NULL;
        return temp;
}

// A utility function to do inorder traversal of BST
voidinorder(struct node *root)
{
        if (root != NULL)
        {
                inorder(root->left);
                printf("%d ", root->key);
                inorder(root->right);
        }
}


struct node* insert(struct node* root, int key)
{
        /* If the tree is empty, return a new node */
        if (root == NULL)
```

```
                returnnewNode(key);


        if (key < root->key)
        root->left = insert(root->left, key);
        else
                root->right = insert(root->right, key);

        /* return the (unchanged) node pointer */
        return root;
}


struct node * Findmin(struct node* node)
{
        struct node* current = node;

        /* loop down to find the leftmost leaf */
        while (current->left != NULL)
                current = current->left;
        return current;
}

struct node* deletion(struct node* root, int key)
{
        // base case
        if (root == NULL) return root;

        // If the key to be deleted is smaller than the root's key,
        // then it lies in left subtree
        if (key < root->key)
                root->left = deletion(root->left, key);

        // If the key to be deleted is greater than the root's key,
        // then it lies in right subtree
        else if (key > root->key)
                root->right = deletion(root->right, key);
```

```
        else
        {
                // node with only one child or no child
                if (root->left == NULL)
                {
                        struct node *temp = root->right;
                        free(root);
                        return temp;
                }
                else if (root->right == NULL)
                {
                        struct node *temp = root->left;
                        free(root);
                        return temp;
                }

                // smallest element from thee right subtree
                struct node* temp = Findmin(root->right);

                // Copy the inorder successor's content to this node
                root->key = temp->key;

                // Delete the inorder successor
                root->right = deletion(root->right, temp->key);
        }
        return root;
}
// main program
void main()
{
        intdata,ch;
        struct node *root = NULL;
        while(1)
        {
                printf("\nenetr 1.insertion 2.display 3.deletion. 4 exit\n");
                printf("enter your choice\n");
                scanf("%d",&ch);
                switch(ch)
```

```
                {

                        case 1: printf("\nenter the value to be inserted\n");
                                scanf("%d",&data);
                                root = insert(root, data);
                                break;
                        case 2: printf("\nInorder traversal of the given tree is: \n");
                                inorder(root);
                                break;

                        case 3: printf("\nenter the value to be deleted\n");
                                scanf("%d",&data);
                                root = deletion(root, data);
                                break;
                        case 4: exit(0);

                }

        }
}
```

OUTPUT:

```
enetr 1.insertion 2.display 3.deletion. 4 exit
enter your choice
1
enter the value to be inserted
20
enetr 1.insertion 2.display 3.deletion. 4 exit
enter your choice
1
enter the value to be inserted
25
enetr 1.insertion 2.display 3.deletion. 4 exit
enter your choice
1
enter the value to be inserted
15
enetr 1.insertion 2.display 3.deletion. 4 exit
enter your choice
1
```

enter the value to be inserted
12
enetr 1.insertion 2.display 3.deletion. 4 exit
enter your choice
1
enter the value to be inserted
16

enetr 1.insertion 2.display 3.deletion. 4 exit
enter your choice
2
enter the value to be deleted

15
enetr 1.insertion 2.display 3.deletion. 4 exit
enter your choice
2
Inorder traversal of the given tree is: 12 16 20 24 25 30

enetr 1.insertion 2.display 3.deletion. 4 exit
enter your choice
4

Experiment 6

Aim: C Program to illustrate insertion, deletion and rotations on AVL Trees

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

// An AVL tree node
struct node
{
	int key;
	struct node *left;
	struct node *right;
	int height;
};

// A utility function to get maximum of two integers
int max(int a, int b);

// A utility function to get height of the tree
int height(struct node *N)
{
	if (N == NULL)
		return 0;
	return N->height;
}

// A utility function to get maximum of two integers
int max(int a, int b)
{
	return (a > b)? a : b;
}

/* Helper function that allocates a new node with the given key and
	NULL left and right pointers. */
struct node* newNode(int key)
{
	struct node* node = (struct node*)
	malloc(sizeof(struct node));
```

```
        node->key = key;
        node->left = NULL;
        node->right = NULL;
        node->height = 1; // new node is initially added at leaf
        return(node);
}

// A utility function to right rotate subtree rooted with y
// See the diagram given above.
struct node *rightRotate(struct node *y)
{
        struct node *x = y->left;
        struct node *T2 = x->right;

        // Perform rotation
        x->right = y;
        y->left = T2;

        // Update heights
        y->height = max(height(y->left), height(y->right))+1;
        x->height = max(height(x->left), height(x->right))+1;

        // Return new root
        return x;
}

// A utility function to left rotate subtree rooted with x
// See the diagram given above.
struct node *leftRotate(struct node *x)
{
        struct node *y = x->right;
        struct node *T2 = y->left;

        // Perform rotation
        y->left = x;
        x->right = T2;

        // Update heights
```

```
        x->height = max(height(x->left), height(x->right))+1;
        y->height = max(height(y->left), height(y->right))+1;

        // Return new root
        return y;
}

// Get Balance factor of node N
intgetBalance(struct node *N)
{
        if (N == NULL)
                return 0;
        return height(N->left) - height(N->right);
}

struct node* insert(struct node* node, int key)
{
        /* 1. Perform the normal BST rotation */
        if (node == NULL)
                return(newNode(key));

        if (key < node->key)
                node->left = insert(node->left, key);
        else
                node->right = insert(node->right, key);

        /* 2. Update height of this ancestor node */
        node->height = max(height(node->left), height(node->right)) + 1;

        /* 3. Get the balance factor of this ancestor node to check whether
        this node became unbalanced */
        int balance = getBalance(node);

        // If this node becomes unbalanced, then there are 4 cases

        // Left Left Case
        if (balance > 1 && key < node->left->key)
                returnrightRotate(node);
```

```
        // Right Right Case
        if (balance < -1 && key > node->right->key)
                returnleftRotate(node);

        // Left Right Case
        if (balance > 1 && key > node->left->key)
        {
                node->left = leftRotate(node->left);
                returnrightRotate(node);
        }

        // Right Left Case
        if (balance < -1 && key < node->right->key)
        {
                node->right = rightRotate(node->right);
                returnleftRotate(node);
        }

        /* return the (unchanged) node pointer */
        return node;
}

// A utility function to print preorder traversal of the tree.
// The function also prints height of every node
voidpreOrder(struct node *root)
{
        if(root != NULL)
        {
                printf("%d ", root->key);
                preOrder(root->left);
                preOrder(root->right);
        }
}

/* Drier program to test above function*/
void main()
{
        intdata,ch;
```

```
        struct node *root = NULL;
        while(1)
        {
                printf("\nenetr 1.insertion 2.display 3.deletion. 4 exit\n");
                printf("enter your choice\n");
                scanf("%d",&ch);
                switch(ch)
                {

                        case 1: printf("\nenter the value to be inserted\n");
                                scanf("%d",&data);
                                root = insert(root, data);
                                break;
                        case 2: printf("\n preorder traversal of the given tree is: \n");
                                preOrder(root);
                                break;

                        /*case 3: printf("\nenter the value to be deleted\n");
                                scanf("%d",&data);
                                root = deletion(root, data);
                                break;*/
                        case 4: exit(0);


                }
        }
}
```

OUTPUT:

```
enetr 1.insertion 2.display 3.deletion. 4 exit
enter your choice
1
20

enetr 1.insertion 2.display 3.deletion. 4 exit
enter your choice
1
```

15

enetr 1.insertion 2.display 3.deletion. 4 exit
enter your choice
1
12

enetr 1.insertion 2.display 3.deletion. 4 exit
enter your choice
2
preorder traversal of the given tree is:

15 12 20

enetr 1.insertion 2.display 3.deletion. 4 exit
4

Experiment 7

Aim:Write a 'C' program to implement Breadth first search.

```
#include <stdio.h>
#include<conio.h>
voidcreategraph();
voidbfs(); void display();
int g[10][10],n;
void main()
{
        int v;
        clrscr();
        creategraph();
        printf("starting vertex is");
        scanf("%d",&v);
        bfs(v);
        getch();
}
voidcreategraph()
{
        inti,j;
        printf("enter the number of nodes");
        scanf("%d",&n);
        for(i=0;i",v);
        while(f<=r)
        {
                v=q[f];
                f++;
                for(i=0;i",i);
                visited[i]=1;
                q[++r]=i;
        }
}
```

Output:

```
Enter no of nodes 3
Edge present between A&A  0
Edge present between A&B  1
Edge present between A&C  1
Edge present between B&A  1
Edge present between B&B  0
```

Edge present between B&C 1
Edge present between C&A 1
Edge present between C&B 1
Edge present between C&C 0

Enter vertex from abopve graph 0

	A	B	C
A	**0**	**1**	**1**
B	**1**	**0**	**1**
C	**1**	**1**	**0**

Aim:Write a 'C' program to implement Depth first search.

```
#include<stdio.h>
#include <conio.h>
voidcreategraph();
voiddfs();
void display();
int g[10][10],n;
void main()
{
        int v;
        clrscr();
        creategraph();
        printf("starting vertex is");
        scanf("%d",&v);
        dfs(v);
        getch();
}
voidcreategraph()
{
        inti,j;
        printf("enter the number of nodes");
        scanf("%d",&n);
        for(i=0;i=0)
        {
                v=st[top];
                top--;
                if(visited [v]==0)
                {
                        printf("%d->",v);
                        visited [v]=1;
                 }
                for(i=n-1;i>=0;i--)
                {
                        if(g[v][i]!=0 && visited[i]==0)
                        {
                                st[++top]=i;
                         }
                }}}
```

Output:

```
Enter no.of nodes 3
Edge present b/w A&A   0
```

Edge present b/w A&B 1
Edge present b/w A&C 1
Edge present b/w A&D 0
Edge present b/w B&A 1
Edge present b/w B&B 0
Edge present b/w B&C 0
Edge present b/w B&D 1
Edge present b/w C&A 1
Edge present b/w C&B 0
Edge present b/w C&C 0
Edge present b/w C&D 0
Edge present b/w D&A 0
Edge present b/w D&B 1
Edge present b/w D&C 0
Edge present b/w D&D 0
Enter vertex from above graph
0
A->B->D->C->

	A	B	C	D
A	**0**	**1**	**1**	**0**
B	**1**	**0**	**0**	**1**
C	**1**	**0**	**0**	**0**
D	**0**	**1**	**0**	**0**

Experiment -8:

Write a program to implement hash table using linear and quadratic probing

```
#include <stdio.h>
#include<stdlib.h>
#define TABLE_SIZE 10

int h[TABLE_SIZE]={NULL};

void insert()
{

 int key,index,i,flag=0,hkey;
 printf("\nenter a value to insert into hash table\n");
 scanf("%d",&key);
 hkey=key%TABLE_SIZE;
 for(i=0;i<TABLE_SIZE;i++)
   {

    index=(hkey+i)%TABLE_SIZE;

    if(h[index] == NULL)
    {
      h[index]=key;
       break;
    }

   }

   if(i == TABLE_SIZE)

    printf("\nelement cannot be inserted\n");
}
void search()
{

 int key,index,i,flag=0,hkey;
 printf("\nenter search element\n");
```

```
 scanf("%d",&key);
 hkey=key%TABLE_SIZE;
 for(i=0;i<TABLE_SIZE; i++)
 {
   index=(hkey+i)%TABLE_SIZE;
   if(h[index]==key)
   {
     printf("value is found at index %d",index);
     break;
   }
 }
 if(i == TABLE_SIZE)
   printf("\n value is not found\n");
}
void display()
{

 int i;

 printf("\nelements in the hash table are \n");

 for(i=0;i< TABLE_SIZE; i++)

 printf("\nat index %d \t value =  %d",i,h[i]);

}
main()
{
   int opt,i;
   while(1)
   {
      printf("\nPress 1. Insert\t 2. Display \t3. Search \t4.Exit \n");
      scanf("%d",&opt);
      switch(opt)
      {
         case 1:
            insert();
            break;
```

```
        case 2:
           display();
           break;
        case 3:
           search();
           break;
        case 4:exit(0);
      }
   }
}
```

Output

```
Press 1. Insert    2. Display    3. Search    4.Exit
1

enter a value to insert into hash table
12

Press 1. Insert    2. Display    3. Search    4.Exit
1

enter a value to insert into hash table
13

Press 1. Insert    2. Display    3. Search    4.Exit
1

enter a value to insert into hash table
22

Press 1. Insert    2. Display    3. Search    4.Exit
2

elements in the hash table are

at index 0     value =  0
at index 1     value =  0
```

```
at index 2      value =  12
at index 3      value =  13
at index 4      value =  22
at index 5      value =  0
at index 6      value =  0
at index 7      value =  0
at index 8      value =  0
at index 9      value =  0
Press 1. Insert     2. Display     3. Search     4.Exit
3

enter search element
12
value is found at index 2
Press 1. Insert     2. Display     3. Search     4.Exit
2 3

enter search element
23

 value is not found

Press 1. Insert     2. Display     3. Search     4.Exit
```

Hashing using quadratic probing:

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

/* to store a data (consisting of key and value) in hash table array */
struct item
{
   int key;
   int value;
};

/* each hash table item has a flag (status) and data (consisting of key and value) */
struct hashtable_item
{

   int flag;
   /*
    * flag = 0 : data does not exist
    * flag = 1 : data exists at given array location
    * flag = 2 : data was present at least once
    */
   struct item *data;

};

struct hashtable_item *array;
int size = 0;
int max = 10;

/* this function returns corresponding index of the given key */
int hashcode(int key)
{
   return (key % max);
}

/* this  function initializes the hash table array */
void init_array()
{
   int i;
   for (i = 0; i < max; i++)
     {
               array[i].flag = 0;
```

```
                array[i].data = NULL;
    }
}

/* this function inserts an element in the hash table */
void insert(int key, int value)
{
    int index = hashcode(key);
    int i = index;
    int h = 1;
    struct item *new_item = (struct item*) malloc(sizeof(struct item));
    new_item->key = key;
    new_item->value = value;

    /* probing through the array until an empty space is found */
    while (array[i].flag == 1)
      {
            if (array[i].data->key == key)
          {
                    /* case when already present key matches the given key */
                    printf("\n This key is already present in hash table, hence updating i
t's value \n");
                    array[i].data->value = value;
                    return;

            }
            i = (i + (h * h)) % max;
            h++;
            if (i == index)
        {
            printf("\n Hash table is full, cannot add more elements \n");
            return;
            }
    }
    array[i].flag = 1;
    array[i].data = new_item;
    printf("\n Key (%d) has been inserted\n", key);
    size++;
}
/* to remove an element form the hash table array */
void remove_element(int key)
{
    it index = hashcode(key);
```

```
    int i = index;
    int h = 1;
    /* probing through the hash table until we reach at location where there had not been
an element even once */
    while (array[i].flag != 0)
      {
            if (array[i].flag == 1  &&  array[i].data->key == key)
          {

                  /* case where data exists at the location and its key matches to the
given key */
                  array[i].flag = 2;
                  array[i].data = NULL;
                  size--;
                  printf("\n Key (%d) has been removed \n", key);
                  return;

            }
            i = (i + (h * h)) % max;
            h++;
            if (i == index)
            {
                  break;
            }
    }
    printf("\n Key does not exist \n");
}
/* to display the contents of hash table */
void display()
{
    int i;
    for(i = 0; i < max; i++)
      {
            if (array[i].flag != 1)
          {
                  printf("\n Array[%d] has no elements \n", i);
            }
            else
          {
                  printf("\n Array[%d] has elements \n  %d (key) and %d (value) \n", i
, array[i].data->key, array[i].data->value);
            }
    }}
```

```
int size_of_hashtable()
{
   return size;
}

void main()
{
   int choice, key, value, n, c;
   clrscr();
 array = (struct hashtable_item*) malloc(max * sizeof(struct hashtable_item*));
   init_array();
   do {
            printf("Implementation of Hash Table in C with Quadratic Probing.\n\n");
            printf("MENU-: \n1.Inserting item in the Hash table"
                  "\n2.Removing item from the Hash table"
                  "\n3.Check the size of Hash table"
                  "\n4.Display Hash table"
                 "\n\n Please enter your choice-:");
            scanf("%d", &choice);
            switch(choice)
          {
            case 1:
                printf("Inserting element in Hash table \n");
                printf("Enter key and value-:\t");
                scanf("%d %d", &key, &value);
                insert(key, value);
                break;
            case 2:
                printf("Deleting in Hash table \n Enter the key to delete-:");
                scanf("%d", &key);
                remove_element(key);
                break;
            case 3:
                n = size_of_hashtable();
                printf("Size of Hash table is-:%d\n", n);
                break;
            case 4:
                display();
                break;
            default:
                 printf("Wrong Input\n");
            }printf("\n Do you want to continue-:(press 1 for yes)\t");
            scanf("%d", &c);
```

```
        }while(c == 1);
        getch();
    }
```

Output:

```
Implementation of Hash Table in C with Quadratic Probing
MENU-:
1. Inserting item in the Hash table
2. Removing item from the Hash table
3. Check the size of Hash table
4. Display Hash table

Please enter your choice-: 3
Size of hash table is-: 0

Do you want to continue-:(press 1 for yes) 1
Implementation of Hash Table in C with Quadratic Probing
MENU-:
1. Inserting item in the Hash table
2. Removing item from the Hash table
3. Check the size of Hash table
4. Display Hash table

Please enter your choice-: 1
Inserting element in Hash table
Enter key and value-:   12     10

 Key (12) has been inserted

Do you want to continue-:(press 1 for yes) 1
Implementation of Hash Table in C with Quadratic Probing
MENU-:
1. Inserting item in the Hash table
2. Removing item from the Hash table
3. Check the size of Hash table
4. Display Hash table

Please enter your choice-: 1
Inserting element in hash table
Enter key and value-:   122   4

 Key (122) has been inserted

Do you want to continue-:(press 1 for yes) 1
```

Implementation of Hash Table in C with Quadratic Probing
MENU-:
1. Inserting item in the Hash table
2. Removing item from the Hash table
3. Check the size of Hash table
4. Display Hash table

Please enter your choice-: 1
Inserting element in hash table
Enter key and value-: 82 5

Key (82) has been inserted

Do you want to continue-:(press 1 for yes) 1
Implementation of Hash Table in C with Quadratic Probing
MENU-:
1. Inserting item in the Hash table
2. Removing item from the Hash table
3. Check the size of Hash table
4. Display Hash table

Please enter your choice-: 3
Size of hash table is-: 3

Do you want to continue-:(press 1 for yes) 1
Implementation of Hash Table in C with Quadratic Probing
MENU-:
1. Inserting item in the Hash table
2. Removing item from the Hash table
3. Check the size of Hash table
4. Display Hash table

Please enter your choice-: 4
Array[0] has no elements

Array[1] has no elements

Array[2] has elements-:
12 (key) and 10 (value)

Array[3] has elements-:
122(key) and 4(value)

Array[4] has no elements

Array[5] has no elements

Array[6] has no elements

Array[7] has no elements
82(key) and 5(value)
Array[8] has no elements

Array[9] has no elements

Do you want to continue-:(press 1 for yes) 1
Implementation of Hash Table in C with Quadratic Probing
MENU-:
1. Inserting item in the Hash table
2. Removing item from the Hash table
3. Check the size of Hash table
4. Display Hash table

Please enter your choice-: 2
Deleting in hash table
Enter the key to delete-: 122

Key (122) has been removed

Do you want to continue-:(press 1 for yes) 1
Implementation of Hash Table in C with Quadratic Probing
MENU-:
1. Inserting item in the Hash table
2. Removing item from the Hash table
3. Check the size of Hash table
4. Display Hash table

Please enter your choice-: 2
Deleting in hash table
Enter the key to delete-: 56

This key does not exist

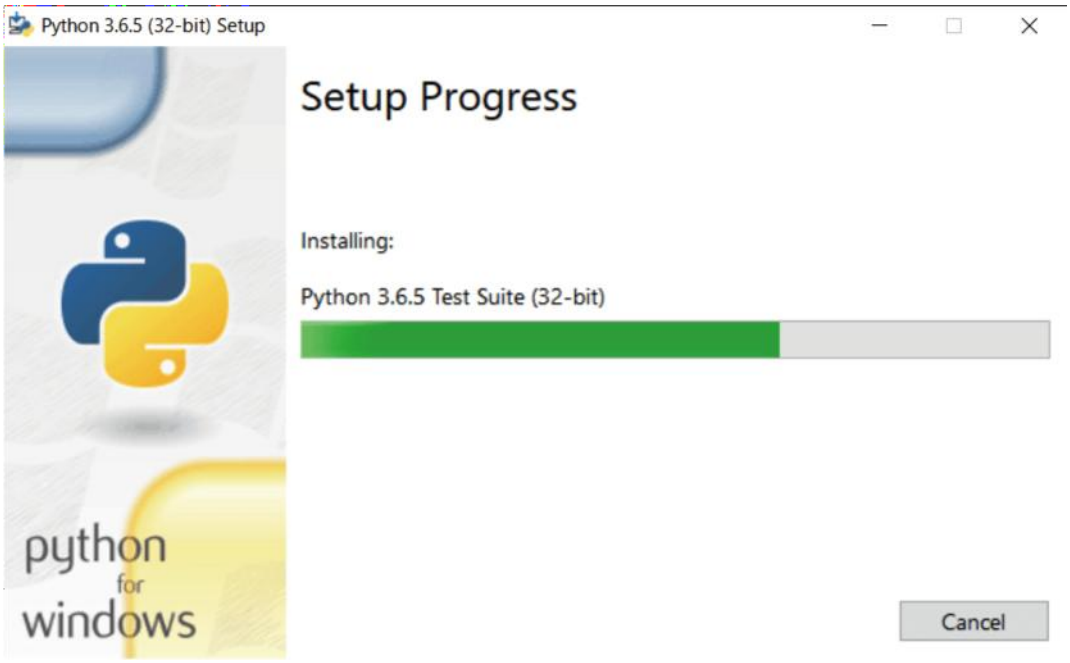
Do you want to continue-:(press 1 for yes) 2

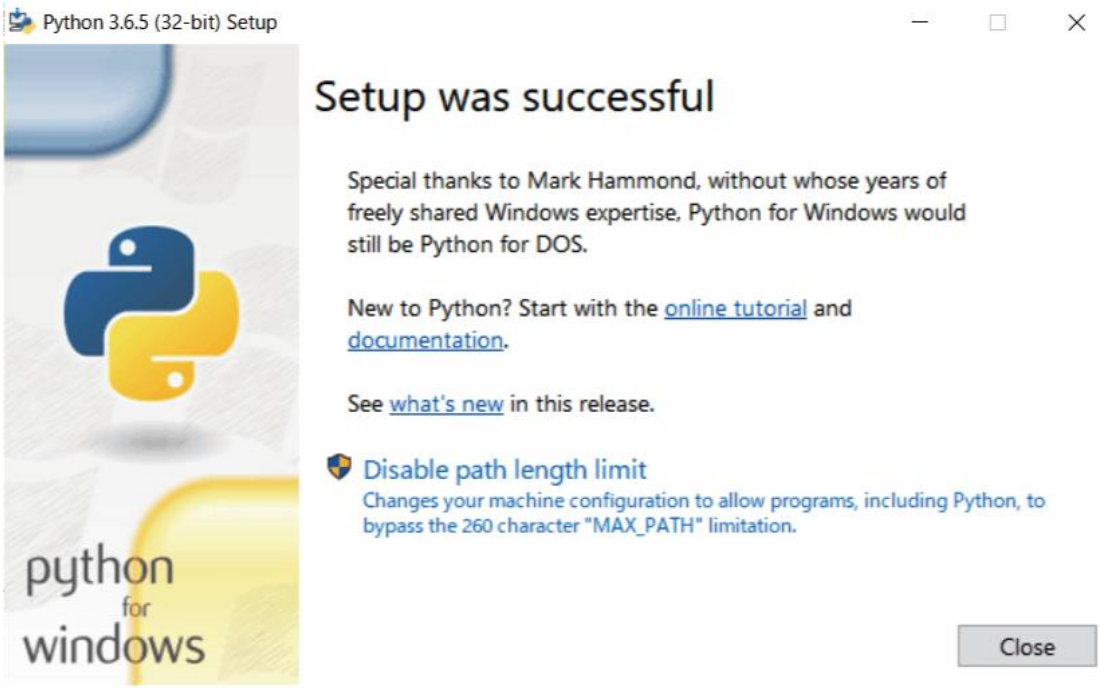
Exercise 1

a) Installation and Environment setup of python.

Here are the steps to install Python on Windows machine.

- Open a Web browser and go to https://www.python.org/downloads/.
- Follow the link for the Windows installer *python-XYZ.msi* file where XYZ is the version you need to install.
- To use this installer *python-XYZ.msi*, the Windows system must support Microsoft Installer
  2.0. Save the installer file to your local machine and then run it to find out if your machine supports MSI.
- Run the downloaded file. This brings up the Python install wizard, which is really easy to use. Just accept the default settings, wait until the install is finished, and you are done.


Python 3.6.5 (32-bit) Setup
Setup Progress
Installing:
Python 3.6.5 Test Suite (32-bit)
python
for
windows
Cancel


Python 3.6.5 (32-bit) Setup
Setup was successful
Special thanks to Mark Hammond, without whose years of freely shared Windows expertise, Python for Windows would still be Python for DOS.
New to Python? Start with the online tutorial and documentation.
See what's new in this release.
Disable path length limit
Changes your machine configuration to allow programs, including Python, to bypass the 260 character "MAX_PATH" limitation.
python
for
windows
Close

b) Write a program to implement basic data types

```
a = 5
print(a, "is of type", type(a))
a = 2.0
print(a, "is of type", type(a))
a = 1+2j
print(a, "is complex number?", isinstance(1+2j,complex))
```

Output:

```
5 is of type <class 'int'>
2.0 is of type <class 'float'>
(1+2j) is complex number? True
```

c) Write a program to demonstrate operators and expressions

i) Program to implement Arithmetic operators

```
x = 15
y = 4
print('x + y =',x+y)
print('x - y =',x-y)
print('x * y =',x*y)
print('x / y =',x/y)
print('x // y =',x//y)
print('x ** y =', x**y)
```

Output:

```
x + y = 19
x - y = 11
x * y = 60
x / y = 3.75
x // y = 3
x ** y = 50625
```

ii) Program to implement Comparison Operators

```
x = 10
y = 12
print('x > y  is',x>y)
print('x < y  is',x<y)
print('x == y is',x==y)
print('x != y  is',x!=y)
```

Output:

```
x > y is False
x < y is True
x == y is False
x != y is True
```

iii) Program to implement Logical Operators

```
x =  True
y = False
print('x and y is',x and y)
print('x or y is',x or y)
print('not x is',not x)
```

Output:

```
x and y is False
x or y is True
not x is False
```

iv) Program to implement Identity operators

```
x1  =  5
y1  =  5
x2  =  'Hello'
y2  =  'Hello'
x3  =  [1,2,3]
y3   =  [1,2,3]
print(x1 is not y1)
print(x2 is y2)
print(x3 is y3)
```

Output:

```
False
True
False
```

v) Program to implement Membership operators

```
x = 'Hello world'
y = {1:'a',2:'b'}
print('H' in x)
print('hello' not in x)
print(1 in y)
print('a' in y)
```

Output:

```
True
True
True
False
```

d) Parameter passing techniques

i) Program to implement functions

```
def myFun():
    "myFunction DocString"  #optional
    print('This is my python function example.')
    print('End of MyFun definition.')
print('Function Example Program:')
myFun()
print('Program Ends..')
```

Output:

```
Function Example Program:
This is my python function example.
End of MyFun definition.
Program Ends..
```

Function Arguments

We can call a function by using the following types of formal arguments −

1. Required arguments
2. Keyword arguments
3. Default arguments
4. Variable-length arguments

Note: In python, **all the functions** are **called by reference,** i.e., all the changes made to the reference inside the function revert back to the original value referred by the reference.

ii) Program to implement Required arguments

```
def keyarg(name, age):
      print("Name = " , name)
      print("Age = ", age )
keyarg("Eeswar", 20)
```

Output:

```
Name = Eeswar
Age = 20
```

iii) Program to implement Keyword arguments

```
def keyarg(name, age):
      print("name = " , name)
      print("age = ", age )
keyarg(age=20, name="Eeswar")
```

Output:

```
name =
Eeswar age =
20
```

iv) Program to implement Default arguments

```
def defaultarg(name, age=18):
      print("name = " , name)
      print("age = ", age )
defaultarg (name= "Eeswar", age=20)
defaultarg (name="Banala")
```

Output:

```
name =  Eeswar
age  =  20
name =  Banala
age  =  18
```

v) Program to implement Variable-length arguments

```
def vararg(name, *sub):
    print("name = ", name)
    print("sub = ", sub)
vararg('Eeswar', 'Python')
vararg('Banala', 'Python', 'Data Structures','Mathematics')
```

Output:

```
name =  Eeswar
sub =  ('Python',)
name =  Banala
sub =  ('Python', 'Data Structures', 'Mathematics')
```

Exercise 2

a) Write a Program to implement

i. Packages ii. Modules iii. Built-in Functions

i) Program and Steps for creating packages in Python:

1. Create a directory with name **PackageDemo** in path **/home**.

2. Create a python source file with name **It.py** and **Cse.py** in the path **/home/PackageDemo**

/home/PackageDemo/It.py

```
def ItFun():
    print('iam in IT module')
```

/home/PackageDemo/Cse.py

```
def  CseFun():
    print('iam in CSE module')
```

3. Now, the directory **PackageDemo** which we have created in the first step contains two python modules. To make this directory a package, we need to include one more file here, that is init .py which contains the import statements of the modules defined in this directory.

/home/PackageDemo /_ _init_ _.py

```
from PackageDemo.Cse import CseFun
from PackageDemo.It import ItFun
```

4. Now, the directory **PackageDemo** has become the package containing two python modules. Here we must notice that we must have to create _ _init_ _.py inside a directory to convert this directory to a package.

5. To use the modules defined inside the package PackageDemo, we must have to import this in our python source file. Let's create a simple python source file at our home directory **(/home)** which uses the modules defined in this package.

/home /TestPackage.py

```
import PackageDemo as p
p.CseFun()
p.ItFun()
```

Output:

```
iam in IT module
iam in CSE module
```

ii) Programs to implement Modules

FunParam.py

```
def add(a,b):
    return (a+b)
def sub(a,b):
    return b-a
def mul(a,b):
    return a*b
def div(a,b):
    return a/b
```

ModuleFrom.py

```
from FunParam import add,sub
sum=add(10,20)
print("Sum = ",sum)
diff=sub(10,20)
print("difference =  ",diff)
```

Output:

```
Sum =  30
difference  =  10
```

iii) Program to implement Built-in Functions

```
print('Absolute of -10.589 = ',abs(-10.589))
print("Binary Equalent of 16", bin(16))
print("Quotient and remainder of 10/5 = " , divmod(10,5))
print("float value of 10 = ", float(10))
print("int value of 10.2 =", int(10.2))
print("Maximum of 10,20,30 = ", max(10,20,30))
print("Minimum of 10,20,30 = ", min(10,20,30))
print("2 power 4 = ", pow(2,4))
import math
print("Square root of 25 = " ,math.sqrt(25))
```

Output:

```
Absolute of -10.589 =  10.589
Binary Equalent of 16 0b10000
Quotient and remainder of  10/5 =  (2, 0)
float value of 10 =  10.0
int value of 10.2 = 10
Maximum of 10,20,30 =  30
Minimum of 10,20,30 =  10
2 power 4 =  16
Square root of 25 =  5.0
```

b) Write a Program to implement

i. List ii. Tuple iii. Dictionaries

i) Implement Stack Data Structure using List Data Type.

```
stack=[]
print("Stack Implementation")
while(1):
    op=int(input("1.Push 2.Pop 3.Display 4.Exit \nEnter your choice: "))
    if(op==1):
        n=input("enter a value:")
        stack.append(n)
    elif(op==2):
        print("\npoped element is ",stack[-1])
        stack.pop()
    elif(op==3):
        for index,i in enumerate(stack):
            print(i,"is at index of ",index )
    elif(op==4):
        break
    else:
        print("Wrong Option")
```

Output:

```
Stack Implementation
1.Push 2.Pop 3.Display 4.Exit
Enter your choice: 1
enter a value:10
1.Push 2.Pop 3.Display 4.Exit
Enter your choice: 1
enter a value:20
1.Push 2.Pop 3.Display 4.Exit
Enter your choice: 1
enter a value:30
1.Push 2.Pop 3.Display 4.Exit
Enter your choice: 3
10 is at index of 0
20 is at index of 1
30 is at index of 2
1.Push 2.Pop 3.Display 4.Exit
Enter your choice: 2
poped element is 30
1.Push 2.Pop 3.Display 4.Exit
Enter your choice: 3
10 is at index of 0
20 is at index of 1
1.Push 2.Pop 3.Display 4.Exit
Enter your choice: 4
```

ii) Program to Implement Tuples

```
# vowels tuple
vowels = ('a', 'e', 'i', 'o', 'i', 'o', 'e', 'i', 'u')

# count element 'i'
count = vowels.count('i')

# print count
print('The count of i is:', count)

# count element 'p'
count = vowels.count('p')

# print count
print('The count of p is:', count)

#index of element 'a'
print('Index of element a in tuple is:',vowels.index('a'))
```

Output:

```
The count of i is: 3
The count of p is: 0
Index of element a in tuple is: 0
```

iii) Program to implement Dictionaries

```
my_dict = {'name':'Eeswar', 'age': 16}
print('Dictionary = ' ,my_dict)
print('Name = ' , my_dict['name'])
print('Age = ' , my_dict.get('age'))

# update value
my_dict['age'] = 18
print('Dictionary after upadte is : ', my_dict)

# add item
my_dict['address'] = 'Hyderabad'
print('Dictionary after adding addres is : ',my_dict)

# remove a particular item
my_dict.pop('age')
print('Dictionary after removing age is : ',my_dict)

# remove all items
my_dict.clear()
print("Dictionary after removing all items : ",my_dict)
```

Output:

```
Dictionary = {'name': 'Eeswar', 'age': 16}
Name = Eeswar
Age = 16
Dictionary after upadte is : {'name': 'Eeswar', 'age': 18}
Dictionary after adding addres is : {'name': 'Eeswar', 'age': 18,
'address': 'Hyderabad'}
Dictionary after removing age is : {'name': 'Eeswar', 'address':
'Hyderabad'}
Dictionary after removing all items : {}
```

c) Programs on Stings and String Operations

1. Program to implement Traversing a String

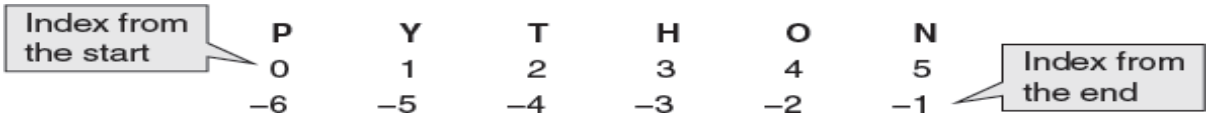
```
s1="Eeswar"
index=0
for  i  in s1:
     print("s1[",index,"]=",i)
     index+=1
```

Output:

```
s1[  0  ]=  E
s1[  1  ]=  e
s1[  2  ]=  s
s1[  3  ]=  w
s1[  4  ]=  a
s1[ 5 ]=  r
```

2. Implementation of Slice Operator

A substring of a string is called a slice. The slice operation is used to refer to sub-parts of sequences and strings. You can take subset of string from original string by using [ ] operator also known as slicing operator.




```
s1="EswarBanala"
print(s1)                          #EswarBanala
print("s1[0:5]=",   s1[0:5])      #Eswar
print("s1[5:]=",   s1[5:])        #Banala
print("s1[:5]=",s1[:5])           #Eswar
print("s1[:]=",s1[:])             #EswarBanala
```

3. Implementation of Stride in Slicing Strings:

In the slice operation, you can specify a third argument as the stride, which refers to the number of characters to move forward after the first character is retrieved from the string.

- By default the value of stride is 1, so in all the above examples where we had not specified the stride, it used the value of 1 which means that every character between two index numbers is retrieved.

```
s1="EswarBanala"
print(s1)                                  #EswarBanala
print("s1[0:5]   =   ",s1[0:5])            #Eswar
print("s1[0:5:1]   =   ",s1[0:5:1])        #Eswar
print("s1[0:5:2]   =   ",s1[0:5:2])        #Ewr
print("s1[0:20:3]   =   ",s1[0:20:3])      #Eaal
```

4. Implementation of Concatenating String:

Adding a string with another string.

```
s1="Eeswar"
s2="Banala"
s3=s1+s2
print("Concatenated String is :", s3)
```

Output:

```
Concatenated String is : EeswarBanala
```

5. Implementation of Multiplying a String with a number:

```
s1="Eeswar "
print(s1*4)
```

Output:

```
Eeswar Eeswar Eeswar  Eeswar
```

6. Implementation of Appending a String:

```
s1="Hello "
s2=input("enter your name:")
s1+=s2
s1+=".. Welcome"
print(s1)
```

Output:

```
enter your name:Eeswar
Hello Eeswar.. Welcome
```

d) Programs on Regular Expressions

1. Implementation of match() Function

The match() function matches a pattern to string and returns a Match object if there is a match found else it returns None.

Syntax

re.match (pattern, string)

Examples

```
matchObj=re.match("[E|e][a-z]+",line)      #string starts with E or e
matchObj=re.match("^(E|e)[a-zA-Z]*",line)
matchObj=re.match("[R|S][a-z]+a$",line) #Namse starts with R or S and ends with a
matchObj=re.match("\d",line)     #any single digit matchOb
j=re.match("\D",line)            # Except digits matchObj=
re.match("\d+",line)             # Minimum 1 digit matchOb
j=re.match("\d*",line)           # Zero or n no. of digits
```

Program to accept a 10 digit mobile number

```
import re
line=input("Enter a Mobile number:")

#mobile number starts with 6,7,8,9
matchObj = re.match('([6-9]\d{9})', line)
if matchObj:
     if matchObj.group()==line:
          print(matchObj.group())
          print ("Matched!!")
     else:
          print ("Not matched!!")
else:
    print ("Not matched!!")
```

Output:

```
enter string:9885986909
9885986909
Matched!!
```

Output:

```
Enter a Mobile number:986879
Not  matched!!
```

2. Implementation of The search() Function

- The search() function searches the string for a match, and returns a Match object if there is a match.
- If there is more than one match, only the first occurrence of the match will be returned:
- The search() function in the re module searches for a pattern anywhere in the string.

Syntax

re.search(pattern, string, flags=0)

The syntax is similar to the match() function. The function searches for first occurrence of pattern within a string with optional flags. If the search is successful, a match object is returned and None otherwise.

Program

```
import re
str = "VJIT Hyderabad"
x = re.search("Hyderabad", str)
print(x)
```

Output:

```
<re.Match object; span=(5, 14), match='Hyderabad'>
```

3. Implementation of The findall() and split() Functions

The findall() function returns a list containing all matches of the string.

Syntax:

```
re.findall(Pattern, str)
```

The split() function returns a list without all the matched patterns of the string .

Syntax:

```
re.findall(Pattern, str)
```

Program

```
import re
str = "IT Dept VJIT"
x = re.findall("IT", str)
print(x)
x=re.split("IT",str)
print(x)
```

Output:

```
['IT', 'IT']
['', ' Dept VJ', '']
```

4. Program to implement The sub() function

The sub() function in the re module can be used to search a pattern in the string and replace it with another pattern.

Syntax

re.sub(pattern, repl, string, max=0)

According to the syntax, the sub() function replaces all occurrences of the pattern in string with repl, substituting all occurrences unless any max value is provided. This method returns modified string.

Program

```
import re
s1="Delhi is the Best city in India."
s2=re.sub("Delhi","Hyderabad",s1)
print(s2)
s1="Delhi is the Best city in India. Delhi Delhi"
s2=re.sub("Delhi","Hyderabad",s1,2)
print(s2)
```

Output:

```
Hyderabad is the Best city in India.
Hyderabad is the Best city in India. Hyderabad Delhi
```

Exercise 3

a) Write Programs to implement Classes and Objects

1) Basic Program on Class and Object

```
class Train:
    # class attributes
    name=input("Enter Train Name: ")
    no=input("Enter Train Number: ")
    price=int(input("Enter Price: "))

    # class methods
    def disp(self):
        print("inside the class method..")
    def status(self,source,destination):
        print("Train starts from ",source," and ends at ",destination)

# Creating an object for Train Class
T1=Train()

# Accessing variables
print("T1 Name: ",T1.name)
print("T1 no: ",T1.no)
print("price:  ",T1.price)

# Calling a Class methods
T1.disp()
T1.status('Hyderabad','Tirupati')
```

Output:

```
Enter Train Name: Venkatadri Express
Enter Train Number: 12345
Enter Price: 1500
T1 Name:Venkatadri  Express
T1  no:   12345
price:    1500
inside the class method..
Train  starts  from   Hyderabad  and ends at  Tirupati
```

2) Write a Python program that reads student details: rollno, name and marks in six subjects and computes average marks. Use classes and objects.

```
class Student:
    def_init_(self,r,n,m):
        print("roll no = ", r)
        print("name = " ,n)
        tot=0
        for i in range(len(m)):
            print(m[i])
            tot=tot+m[i]
        print("total= ", tot)
        avg=tot/6
        print("average= ",avg)

roll=input("enter roll : ")
name=input("enter name : ")
marks=[]
print("enter marks : ")
for i in range(1,7):
    marks.append(int(input()))
print(marks)

s1=Student(roll,name,marks)
```

Output:

```
enter roll : 1208
enter name : Eeswar
enter marks :
49
39
40
60
90
57

roll no = 1208
name = Eeswar
49
39
40
60
90
57
total= 335
average= 55.833333333333336
```

b) Write a Program to implement Static and Instance methods

```
class  Shape:
      def rectArea(self,l,b):
            return(l*b)
      @classmethod
      def sqArea(cls,s):
            return(s*s)
      @staticmethod
      def CircleArea(r):
            return(3.14*r*r)
s=Shape()
print("Area of Rectangle = ",s.rectArea(2,4))
#calling a class method
print("Area of Square = ",Shape.sqArea(3))
#calling a static method
print("Area of Circle = ",Shape.CircleArea(2))
```

Output:

```
Area  of Rectangle  =    8
Area  of Square  =    9
Area  of  Circle  =    12.56
```

c) Program to implement Abstract Classes.

```
import  math
class Shape:          #Abstract Class
      def  calculateArea(self):
            raise NotImplementedError()

class  Circle(Shape):
      def_init_(self,r):
            self.radius=r
      def calculateArea(self):
            area = math.pi * math.pow(self.radius,2)
            print("Area of circle = ",area)

class Traingle(Shape):
      def_init_(self,h,b):
            self.height=h
            self.base=b
      def calculateArea(self):
            area = (self.height * self.base)/2
            print("Area of Traingle = ",area)

class Rectangle(Shape):
      def_init_(self,l,w):
            self.lenghth=l
            self.width=w
      def calculateArea(self):
            area = self.lenghth * self.width
            print("Area of Rectangle = ",area)

c1=Circle(2)
t1=Traingle(5,3)
r1=Rectangle(2,4)

c1.calculateArea()
t1.calculateArea()
r1.calculateArea()
```

Output:

```
Area  of  circle  =   12.566370614359172
Area  of  Traingle  =7.5
Area  of  Rectangle  =   8
```

Exercise 4

a) Write a program to compute distance between two points taking input from the user (Pythagorean Theorem)

```
import  math

# Function to calculate distance
def distance(x1 , y1 , x2 , y2):
      dis= math.sqrt(math.pow(x2 - x1, 2) + math.pow(y2 - y1, 2) * 1.0)
      return  dis

#Code
x1=int(input("enter x1: "))
y1=int(input("enter y1: "))
x2=int(input("enter x2: "))
y2=int(input("enter y2: "))
dis=distance(x1,y1,x2,y2)
print("distance=",dis)
```

Output:

```
enter  x1:  5
enter  y1:  4
enter  x2:  7
enter  y2:  8
distance=  4.47213595499958
```

b) Write a program to convert a given decimal number to other base systems

```
def binarynum(num):
    binnum=bin(num)
    print(num," in binary is  ",binnum)

def octnum(num):
    octnum=oct(num)
    print(num," in octal is ",octnum)

def hexnum(num):
    hexnum=hex(num)
    print(num," in Hexal decimal is ",hexnum)

num=int(input("enter a decimal number : "))
binarynum(num)
octnum(num)
hexnum(num)
```

Output:

```
enter a decimal number : 15
15  in  binary  is   0b1111
15  in  octal  is   0o17
15  in Hexal  decimal  is   0xf
```

Exercise 5

a) Write a program to implement Inheritance

1) Program to implement Single Inheritance

```
class  Bank:
      def getroi(self):
            return 10;
class  SBI(Bank):
      def getroi(self):
            return 7;

class ICICI(Bank):
      def  getroi(self):
            return 8;
b1 = Bank()
b2 =  SBI()
b3 =  ICICI()
print("Bank Rate of interest:",b1.getroi())
print("SBI Rate of interest:",b2.getroi())
print("ICICI Rate of  interest:",b3.getroi())
```

Output:

```
Bank Rate of interest: 10
SBI Rate of interest: 7
ICICI Rate of interest: 8
```

2) Program to implement multiple inheritance

```
class Area:
    def getArea(length,breadth):
        a=length*breadth
        return(a)
class  Perimeter:
    def getPerimeter(length,breadth):
        p=2*(length+breadth)
        return(p)
class Rectangle(Area,Perimeter):
    def_init_(self,l,b):
        self.length=l
        self.breadth=b
    def area(self):
        area= Area.getArea(self.length,self.breadth)
        print("Area of Rectangle= ",area)
    def perimeter(self):
        perimeter=Perimeter.getPerimeter(self.length,self.breadth)
        print("Perimeter of Rectangle= ",perimeter)

r1=Rectangle(7,4)
r1.area()
r1.perimeter()
```

Output:

```
Area of Rectangle=  28
Perimeter of Rectangle=  22
```

3) Program to implement Multilevel Inheritance

```
class  Car:
      def vehicleType(self):
            print("VehicleType : car")
class Suziki(Car):
      def  brand(self):
            print("Brand :Suziki")
      def speed(self):
            print("Max Speed : 120 KMPH")
class  SwiftDezire(Suziki):
      def speed(self):
            print("Max Speed : 200 KMPH")

obj1=SwiftDezire()
obj1.vehicleType()
obj1.brand()
obj1.speed()

obj2=Suziki()
obj2.vehicleType()
obj2.brand()
obj2.speed()
```

Output:

```
VehicleType : car
Brand : Suziki
Max Speed : 200 KMPH
VehicleType : car
Brand : Suziki
Max Speed : 120  KMPH
```

4) Program to implement Multi-path Inheritance

```
class  Student:
      def_init_(self,n):
            self.name=n
      def getName(self):
            print("Name :  ",self.name)

class Branch(Student):
      def getBranch(self, dept):
            print("Branch : ", dept)

class Score(Student):
      def getScore(self,avg):
            print("Academic Score : ", avg)

class Result(Branch,Score):
      def getResult(self):
            print('Result'.center(20,'*'))
            self.getName()
            self.getBranch('IT')
            self.getScore(65)

r=Result("Eeswar")
r.getResult()
```

Output:

```
*******Result*******
Name :  Eeswar
Branch :IT
Academic Score :  65
```

b) Write a program to implement Polymorphism

```
class  Parent:
    def   myMethod(self):
        print('Calling parent method')

class Child(Parent):
    def myMethod(self):
        print('Calling child method')

c  =  Child()              # instance  of  child
c.myMethod()               # child calls overridden  method
```

Output:

```
Calling child method
```

Exercise 6

a) Write a program to implement Files

1) Program to implement file attributes

```
file=open("Sample1.txt","r")
print("Name of the file:",file.name)
print("File open status:",file.closed)
print("File opened mode:",file.mode)
print("Content in the file:")
print(file.read())
file.close()
print("File open status:",file.closed)
```

Output:

```
Name of the file: Sample1.txt
File open startus: False
File opened mode: r
Content in the file:
"FILES" DEMONSTRATION IN PYTHON...
File open status: True
```

2) Program to implement file operations

```
import os
f1=open("text2.txt","w")
f1.write("Welcome to python class")
f1.close()
print("New file \"text2.txt\"is created in the path.")
f1=open("text2.txt","a")
f1.write(" Happy Learning")
f1.close()
f1=open("text2.txt","r")
print("Filepointer position when it was opened: ", f1.tell())
f1.seek(3)
print("Filepointer position after seek(): ", f1.tell())
print("Contents in file: " , f1.read())
print("Filepointer position after read(): ", f1.tell())
f1.close()
os.rename("text2.txt","text3.txt")
os.remove("text3.txt")
```

Output:

```
New file "text2.txt"is created in the path.
Filepointer position when it was opened: 0
Filepointer position after seek(): 3
Contents in file: come to python class Happy Learning
Filepointer position after read(): 38
```

b) Write a program to Exception Handling.

```
try:
    num=int(input("Enter numerator: "))
    den=int(input("enter denominator: "))
    q=num/den
    print("Qutioent= ",q)
except ZeroDivisionError:
    print("Denominator is zero")
except ValueError:
    print("Please eneter only digits")
finally:
    print("this is finally block. this will be executed all the time..");
```

Output:

```
Enter numerator: 20
enter denominator: 0
Denominator is zero
this is finally block. this will be executed all the time..
```

Output:

```
Enter numerator: 10
enter denominator: 5
Qutioent= 2.0
this is finally block. this will be executed all the time..
```

Vidya Jyothi Institute of Technology

(Approved by AICTE, New Delhi, Accredited by NAAC, Permanently Affiliated to JNTUH, Hyderabad)

An Autonomous Institution

Aziznagar Gate, ChilkurBalaji Road,

Hyderabad – 500075, Telangana, India

Department of Information Technology

Rubrics for Continuous Assessment

Course: Datastructures & python Programming **Evaluator:** G. Indira Priyadarshini

Date: **Overall Score: 15M**

Objects	5 Out Standing	4 Good	3 Moderate	2 Normal	1 Satisfied	0 Not Satisfied
Observation-5M						
Record-5M						
Execution -5M						

HOD-IT

Head of the Department
Dept. of Information Technology
Vidya Jyothi Institute of Technology
Aziz Nagar Gate, C.B. Road

Lab Assessment:

Continuous Assessment Week Wise	**15M**
Lab Internal Examination	**10M**
Total	**25M**

HOD-IT

Head of the Department
Dept. of Information Technology
Vidya Jyothi Institute of Technology
Aziz Nagar Gate, C.B. Post
Hyderabad - 50 [illegible]

VIDYA JYOTHI INSTITUTE OF TECHNOLOGY

Aziz Nagar Gate, C.B Post, Hyderabad – 500075

DEPARTMENT OF INFORMATION TECHNOLOGY

Continuous Lab Assessment Sheet

Academic Year: 2021-2022
Course Name: DS & PP Lab

Class: II - C'Section

Regulation: R20
Course Code: _____

S.No.	Roll No.	Date : 5/10				Date : 12/10				Date : 26/10				Date : 2/11				Date : 9/11				Date : 16/11				Date : 30/11				Date : 7/11			
		O	R	E	T	O	R	E	T	O	R	E	T	O	R	E	T	O	R	E	T	O	R	E	T	O	R	E	T	O	R	E	T
1	20911A12C1	←	AB		→	5	5	4	14	5	5	5	15	←	AB		→	4	5	5	14	5	5	5	15	3	4	3	10	5	5	4	14
2	20911A12C2	←	AB		→	4	4	5	13	4	5	5	14	←	AB		→	5	5	5	15	4	5	5	14	4	5	5	14	4	5	4	13
3	20911A12C3	←	AB		→	←	AB		→	5	5	4	14	←	AB		→	5	5	4	14	5	5	5	15	5	4	5	14	5	5	4	14
4	20911A12C4	5	4	5	14	5	5	5	15	5	4	4	13	5	5	5	15	5	5	5	15	4	3	5	12	5	5	5	15	4	4	5	13
5	20911A12C5	←	AB		→	5	5	5	15	4	5	4	13	5	5	5	15	4	5	5	14	←	AB		→	5	4	5	14	5	5	5	15
6	20911A12C6	←	AB		→	←	AB		→	5	5	5	15	5	4	5	14	5	5	5	15	5	4	5	14	5	5	5	15	5	4	3	12
7	20911A12C7	5	4	5	14	5	5	5	15	5	4	5	14	5	4	5	14	5	3	2	10	5	3	4	12	5	4	5	14	5	3	3	11
8	20911A12C8	5	3	2	10	5	4	5	14	5	4	5	14	5	5	5	15	5	2	3	10	5	4	3	12	5	4	5	14	5	5	5	15
9	20911A12C9	5	4	5	14	5	5	5	15	5	2	3	10	←	AB		→	5	4	5	14	5	5	4	14	5	3	2	10	5	5	5	15
10	20911A12D0	←	AB		→	5	4	5	14	5	5	5	14	←	AB		→	5	5	4	14	5	3	2	10	5	5	4	14	5	4	5	14
11	20911A12D1	5	5	4	14	5	5	5	15	4	4	4	12	←	AB		→	5	4	5	14	5	5	5	15	5	5	5	15	5	4	5	14
12	20911A12D2	←	AB		→	5	4	5	15	4	5	2	11	5	5	4	14	5	5	4	14	5	5	5	15	6	4	5	14	5	5	5	15
13	20911A12D3	5	4	5	14	5	3	5	13	4	5	3	12	5	4	5	14	5	5	5	15	4	4	5	13	5	5	4	14	5	5	5	15
14	20911A12D4	5	3	2	10	5	4	5	14	5	3	2	10	←	AB		→	5	2	4	11	5	5	5	15	5	5	4	14	5	3	2	10
15	20911A12D5	←	AB		→	5	4	5	14	5	3	2	10	5	3	2	10	5	4	3	12	5	4	5	14	5	5	5	15	5	4	3	12
16	20911A12D6	←	AB		→	5	3	2	10	5	4	5	14	5	3	3	11	5	4	5	14	5	3	3	11	5	4	5	14	5	3	2	10
17	20911A12D7	←	AB		→	←	AB		→	5	5	5	15	5	3	2	10	5	5	3	13	5	4	3	12	5	4	3	12	5	5	4	14
18	20911A12D8	5	5	4	14	5	3	2	10	←	AB		→	5	3	3	11	5	2	3	10	5	4	5	14	5	5	5	15	5	3	3	11
19	20911A12D9	←	AB		→	5	4	3	12	5	4	4	13	5	4	3	12	5	5	5	15	5	4	5	14	5	4	3	12	5	4	3	12
20	20911A12E0	←	AB		→	←	AB		→	←	AB		→	5	5	5	15	5	4	3	12	5	4	5	14	5	5	5	15	5	3	3	11
21	20911A12E1	←	AB		→	5	4	3	12	←	AB		→	5	4	3	12	←	AB		→	5	3	3	11	5	4	5	14	5	4	5	14
22	20911A12E2	←	AB		→	4	4	5	13	4	5	5	14	5	2	3	10	5	4	4	14	5	4	5	14	5	3	3	11	5	4	5	14
23	20911A12E3	5	4	5	14	5	4	3	12	←	AB		→	5	4	3	12	5	4	3	12	5	4	5	14	←	AB		→	5	2	3	10
24	20911A12E4	←	AB		→	5	4	3	12	←	AB		→	5	2	3	10	←	AB		→	5	2	3	10	5	4	5	14	5	4	3	12
25	20911A12E5	←	AB		→	5	5	5	15	5	4	3	12	5	4	4	14	←	AB		→	5	2	3	10	5	2	3	10	5	4	2	11

O - Observation(5M), R - Record(5M), E - Execution(5M), T - Total Marks (15M)

VIDYA JYOTHI INSTITUTE OF TECHNOLOGY

Aziz Nagar Gate, C.B Post, Hyderabad – 500075

DEPARTMENT OF INFORMATION TECHNOLOGY

Continuous Lab Assessment Sheet

S.No.	Roll No.	Date : 5/10				Date : 12/10				Date : 26/10				Date : 2/11				Date : 9/11				Date : 16/11				Date : 30/11				Date : 7/11			
		O	R	E	T	O	R	O	R	E	T	E	T	O	R	E	T	O	R	E	T	O	R	E	T	O	R	E	T	O	R	E	T
26	20911A12E6	5	4	3	12	5	4	5	14	←	AB	→	5	4	3	12	5	5	5	15	~~5~~	~~4~~	A	B →		3				4	5	12	5	4	5	14
27	20911A12E7	←	AB	→	5	4	3	12	←	A	B →	5	2	3	10	5	4	3	12	2	3	5	10		2				3	5	10	5	3	4	12
28	20911A12E8	5	2	3	10	5	4	3	12	←	A	B →	2	3	5	10	4	5	3	12	4	5	2	11		2				5	3	10	2	5	3	10
29	20911A12E9	←	AB	→	←	A	B →	←	A	B →	5	4	4	13	2	5	3	10	4	5	2	11		4				5	2	11	5	2	3	10
30	20911A12F0	←	AB	→	2	5	3	10	←	A	B →	4	5	2	11	←	AB	→	5	2	3	10		5				4	3	12	5	4	4	13
31	20911A12F1	←	AB	→	5	4	2	11	←	A	B →	2	5	3	10	5	4	4	13	2	5	3	10								2	5	3	10
32	20911A12F2	←	AB	→	4	5	2	11	←	A	B →	5	2	3	10	4	5	3	12	5	4	4	13		2				5	3	10	2	3	5	10
33	20911A12F3	←	AB	→	2	5	3	10	←	A	B →	2	3	5	10	4	5	3	12	4	5	2	11		4				5	2	11	5	4	4	13
~~34~~	~~20911A12F4~~	~~←~~	~~AB~~	~~→~~	~~←~~	~~AB~~	~~→~~																										
35	20911A12F5	←	AB	→	←	AB	→	3	2	5	10	5	4	4	13	5	4	4	13	5	4	4	13		4				5	2	11	4	5	4	13
36	20911A12F6	4	3	5	12	5	3	2	10	←	A	B →	5	2	3	10	5	4	4	13	4	5	2	11		2				3	5	10	5	4	3	12
37	20911A12F7	4	3	5	12	←	A	B →	5	2	3	10	←	A	B →	2	5	3	10	5	4	4	13		4				5	2	11	2	3	5	10
38	20911A12F8	3	4	5	12	~~5~~	~~2~~	3	10	←	A	B →	5	4	3	12	4	5	2	11	5	2	3	10		4				5	2	11	3	4	5	12
39	20911A12F9	5	4	3	12	←	A	B →	2	3	5	10	4	5	3	12	4	5	2	11	5	2	3	10		4				5	2	11	2	3	5	10
40	20911A12G0	←	AB	→	5	3	2	10	2	3	5	10	4	5	3	12	4	5	2	11	5	2	3	10		4				5	2	11	2	3	5	10
41	20911A12G1	~~5~~	~~3~~	2	10	5	4	3	12	4	5	3	12	2	5	3	10									2				5	3	10	4	4	5	13
42	20911A12G2	←	AB	→	←	AB	→	5	2	3	10	5	4	3	12	2	3	5	10	4	5	2	11		5				2	3	10	2	3	5	10
43	20911A12G3	←	AB	→	←	AB	→	4	5	3	12	5	4	4	13	4	5	4	13	4	5	2	11		5				2	3	10	3	4	5	12
44	20911A12G4	2	5	3	10	5	4	3	12	2	3	5	10	4	5	4	13	2	~~5~~	3	10														
45	20911A12G5	3	4	5	12	5	4	3	12	3	4	5	12	2	3	5	10	2	3	5	10	4	5	2	11		4			5	2	11	3	4	~~5~~	12
46	20911A12G6	←	AB	→	←	AB	→	3	4	5	12	4	5	3	12	2	5	3	10	4	5	2	11		2				3	5	10	4	5	3	12
47	20911A12G7	←	AB	→	←	AB	→	←	AB	→	5	2	3	10	5	4	3	12	2	5	3	10	4	5	2	11		5			2	3	10	4	5	5	~~4~~
48	20911A12G8	5	4	3	12	5	~~2~~	~~3~~	10	4	5	3	12	5	4	4	13	2	3	5	10	4	5	2	11		2			5	3	10	3	4	5	12
49	20911A12G9	5	2	3	10	4	5	3	12	5	4	4	13	2	3	5	10	3	4	5	12		5				5			2	3	10	5	2	3	10
50	20911A12H0	5	4	3	12	4	5	3	12	5	4	4	13	2	3	5	10	4	5	4	13	2	3	5	10		4			5	3	12	4	5	3	12
51	20911A12H1	3	4	5	12	~~4~~	5	4	~~3~~	←	AB	→	4	5	3	12	2	5	3	10	4	5	2	11		5				2	3	10	4	5	4	13
	20911A12H2																										4			5	3	12	5	4	5	14

O - Observation(5M), **R** - Record(5M), **E** - Execution(5M), **T** - Total Marks (15M)

VIDYA JYOTHI INSTITUTE OF TECHNOLOGY

Aziz Nagar Gate, C.B Post, Hyderabad – 500075

DEPARTMENT OF INFORMATION TECHNOLOGY

Continuous Lab Assessment Sheet

S.No	Roll No.	Date : 5/10				Date : 12/10				Date : 26/10				Date : 2/11				Date : 9/11				Date : 16/11				Date : 30/11				Date : 7/12			
		O	R	E	T	O	R	E	T	O	R	E	T	O	R	E	T	O	R	E	T	O	R	E	T	O	R	E	T	O	R	E	T
52	20911A12H2	5	4	4	13	5	4	5	14	←	A	B	→	4	5	5	14	4	3	2	9	4	5	4	14	5	3	2	10	5	4	5	14
53	20911A12H3	←	A	B	→	←	A	B	→	←	A	B	→	5	4	3	12	5	3	2	10	5	4	2	11	4	5	3	14	4	5	3	12
54	20911A12H4	5	5	4	14	←	A	B	→	5	3	4	12	2	4	5	11	3	2	5	10	5	4	2	11	5	4	5	14	2	4	5	11
55	20911A12H5	4	5	2	11	←	A	B	→	5	3	4	12	5	4	2	11	5	4	5	14	5	5	5	15	5	2	3	10	5	2	3	10
56	20911A12H6	←	A	B	→	←	A	B	→	←	A	B	→	5	2	3	10	5	4	2	11	4	5	2	11	5	3	4	12	4	5	2	11
57	20911A12H7	5	3	4	12	←	A	B	→	5	2	3	10	5	4	2	11	5	2	3	10	5	4	2	11	5	4	5	14	4	3	5	12
58	20911A12H8	4	3	5	12	5	4	5	14	5	5	5	15	5	5	5	15	5	4	5	14	5	4	5	14	5	4	5	14	4	3	4	11
59	20911A12H9	←	A	B	→	5	4	5	14	←	A	B	→	4	3	4	11	5	3	2	10	5	4	5	14	5	5	5	15	5	3	2	10
60	20911A12J0	3	2	5	10	←	A	B	→	5	4	5	14	3	4	4	11	5	3	2	10	5	4	5	14	5	5	5	15	5	4	5	14
61	21915A1210	4	3	5	12	←	A	B	→	5	4	2	11	4	4	3	11	2	3	5	10	5	2	3	10	5	4	5	14	4	4	5	13
62	~~21915A1211~~	2	3	5	10	←	A	B	→	5	3	2	10	3	4	4	11	3	4	4	11	2	3	5	10	5	4	3	12	5	4	3	12
63	21915A1212	5	4	5	14	←	A	B	→	~~4~~	~~A~~	5	14	2	5	3	10	5	4	5	14	5	2	3	10	3	2	5	10	4	3	4	11
64	21915A1213	5	3	2	10	2	4	5	11	5	3	2	10	2	4	5	11	3	5	2	10	2	3	5	10	5	4	3	12	5	2	3	10
65	21915A1214	2	3	5	10	←	A	B	→	←	A	B	→	3	2	5	10	4	4	3	11	4	4	5	13	2	3	5	10	5	4	4	13
66	21915A1215	←	A	B	→	3	2	5	10	←	A	B	→	5	2	3	10	3	4	4	11	5	4	3	12	3	2	5	10	5	2	3	10
67	21915A1216	←	A	B	→	5	3	2	10	←	A	B	→	3	4	4	11	2	3	5	10	3	4	5	12	4	5	3	12	4	5	3	12
68	21915A1217	←	A	B	→	←	A	B	→	2	4	5	11	2	3	5	10	4	4	3	11	2	3	5	10	5	4	3	12	4	4	5	13

O – Observation (5M), **R** - Record(5M), **E** - Execution(5M), **T** - Total Marks (15M)

(Course Instructor)