**Index**

|  |  |
| --- | --- |
| **Unit 1** | Page No. |
| How to run the program in Borland C? | 2 |
| How to run the program in Code::Block? | 3 |
| Simple Programs | 4 |
| Programs using if condition | 5 |
| Loop Programs | 6 |
|  |  |
| **Unit 2** |  |
| Programs related to Arrays | 8 |
| Program 🡪 Different Operations on Array | 9 |
| Program 🡪 Stack | 22 |
| Program 🡪 Queue | 25 |
| Program 🡪 Circular Queue | 27 |
| Programs related to Strings | 29 |
| Programs related to Pointers | 30 |
|  |  |
| **Unit 3** |  |
| Function Programs | 31 |
| Recursive Function Programs | 31 |
| Listing of Programs related to Structures | 32 |
| Programs related to Structures and Union | 38 |
|  |  |
| **Unit 4** |  |
| Programs related to File Handling | 39 |
|  |  |
| **Extra Material** | Extra Material |
| BITS, Bytes and Number Systems | 1 – Number Systems |
| Bitwise Operators | 2- C-Bitwise Operators |
| Pre-processor Directives | 3 - Preprocessor Directives |
| Towards Object Oriented Programming | 4 - Towards Object Oriented Programming |

**Unit 1**

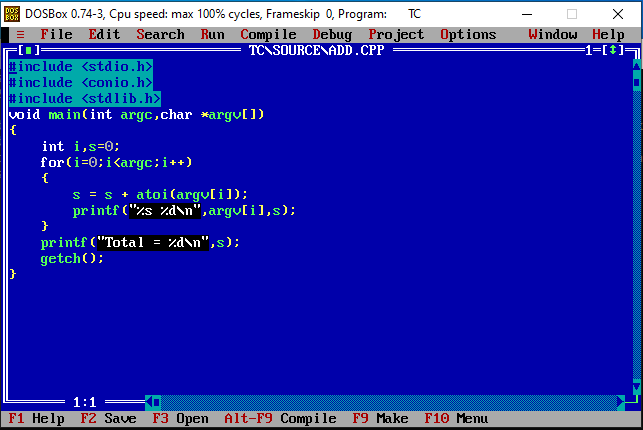
**How to run the program in Borland C?**

Borland C works under DOS and now-a-days, you can't run directly any DOS programs under Windows environment. So, use DOSBox with C compiler. Get some familiarity with DOSBox. Once C compiler is installed, compiling and running the program is really very easy.

Press Alt+F to open the File Menu and then Press N to create a new C program file.

Type your program.

Now, Press Alt + F9 to compile the program. It will list down any errors in another window, otherwise it will flash the message that program has been successfully compiled. Now Press Ctrl + F9 (Alternatively Alt R, R) to run the program.



**How to run the program in Code::Block?**

You can download the software from the following link.

**https://sourceforge.net/projects/codeblocks/files/Binaries/25.03/Windows/codeblocks-25.03mingw-setup.exe/download**

Once successfully installed, you can run the program.

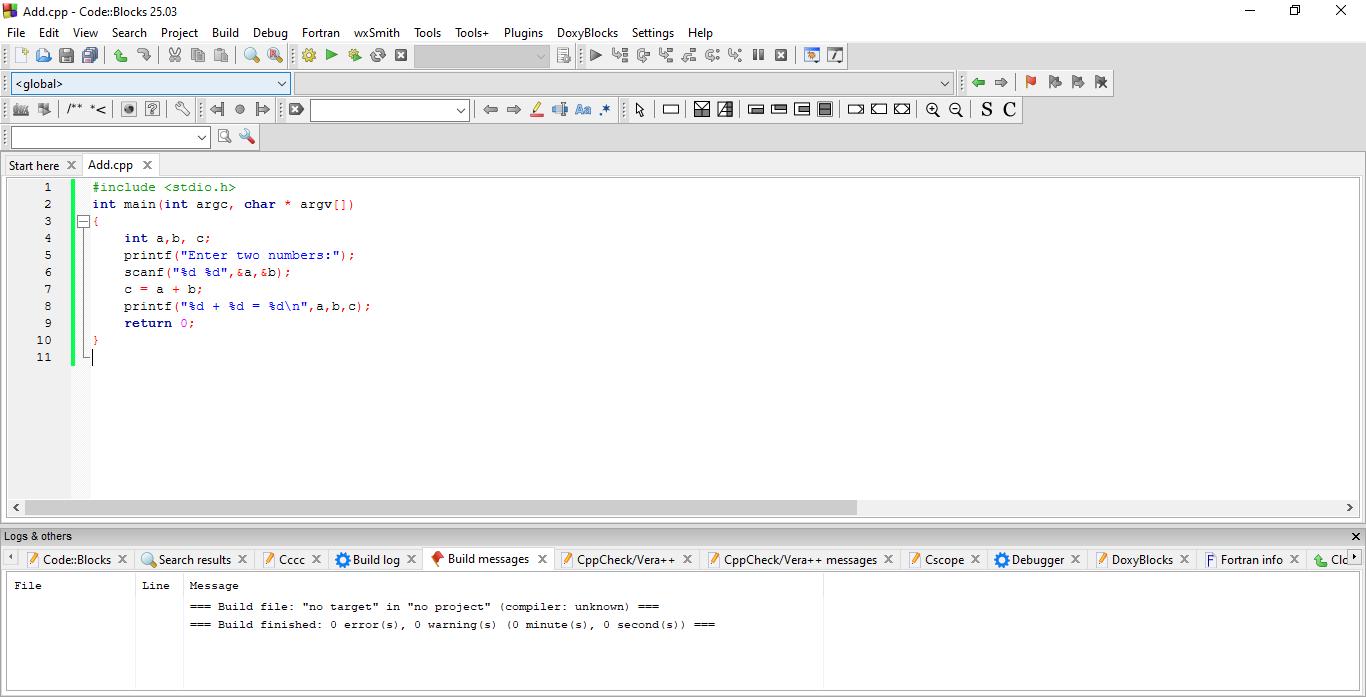
Write your 1st program in CodeBlock:

Press File🡪New🡪file…🡪 C/C++ Source 🡪 Go 🡪 Next 🡪 C++

Provide File Name and then click on Finish.

Type your program.

The screen will look something like below:

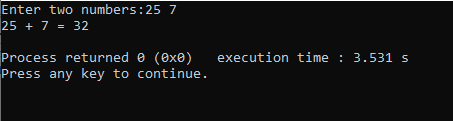


Save the program (Ctrl + S)

Press Ctrl+F9 to Build

Press Ctrl+F10 to Run, if program is successfully compiled.

The output screen will look like as under:



**WRITE C PROGRAMS FOR THE FOLLOWING. (SIMPLE PROGRAMS)**

1. Add two numbers.
2. Subtract two numbers.
3. Multiply two numbers.
4. Divide two numbers.
5. Add, multiply, subtract and divide two numbers.
6. Convert hours into minutes.
7. Convert minutes into hours.
8. Convert dollars into Rs. Where 1 $ = 48 Rs.
9. Convert Rs. into dollars where 1 $ = 48 Rs.
10. Convert dollars into pound where 1 $ = 48 Rs. And 1 pound = 70 Rs.
11. Convert grams into kg.
12. Convert kgs into grams.
13. Convert bytes into KB, MB and GB.
14. Convert celcius into Fahrenheit. F = (9/5 \* C) + 32
15. Convert Fahrenheit into celcius. C = 5/9 \* (F – 32)
16. Calculate interest where I = PRN/100.
17. Calculate area & perimeter of a square. A = L**^**2, P = 4L
18. Calculate area & perimeter of a rectangle. A = L\*B, P = 2 (L+B)
19. Calculate area of a circle. A = 22/7 \* R \* R
20. Calculate area of a triangle. A = H\*L/2
21. Calculate net salary

where net salary = gross salary + allowance – deduction.

Allowances are 10% while deductions are 3% of gross salary.

1. Calculate net sales where net sales = gross sales – 10% discount of gross sales.
2. Calculate average of three subjects along with their total.
3. Swap two values.

**WRITE C PROGRAMS FOR THE FOLLOWING. (USING IF CONDITION)**

1. Find out largest and smallest of two values.
2. Find out largest and smallest of three values.
3. Find out net salary where net salary = gross salary + allowances – deductions.

If gross salary > 10000, allowances are 10%, deductions are 3%

If gross salary > 5000, allowances are 7%, deduction are 2%

1. Find out whether a given no. is divisible by 7 or not.
2. Find out net sales where net sales = gross sales – discount.

If gross sales > 20000, discount is 15%

If gross sales > 10000, discount is 10% otherwise 5%.

1. Calculate total, average of marks of three subjects. Give following grades to the student.

If average >= 70, distinction,

>= 60, first,

>= 50, second,

>= 35, third class,

Otherwise fail.

If student secures < 35 marks in any subject then declare student fail.

**WRITE C PROGRAMS FOR THE FOLLOWING. (USING LOOP)**

1. Print 1st 10 natural nos.
2. Print 1st 10 odd nos.
3. Print 1st 10 even nos.
4. Print 1st n natural nos.
5. Print 1st n odd nos.
6. Print 1st n even nos.
7. Print sum of n natural nos.
8. Print sum of 1st n odd nos.
9. Print sum of 1st n even nos.
10. Print factorial of a given no. e.g. 5! = 5\*4\*3\*2\*1 = 120.
11. Print your name 5 times.
12. Print your name n times.
13. Print sum of all nos. divisible by 13 in the range of 1 and 100.
14. Calculate sum and mean of any 10 values.
15. Calculate sum and mean of any n values.
16. Find out largest and smallest out of 100 nos.
17. Count +ve,-ve and zeroes in 200 values.
18. Find out how many boys and girls are there in a class of 50 students.

Input sex code.

1. Print all integers from 1 to 100, which are divisible by 5
2. Print sum of all integers from 1 to 100, which are divisible by 3.
3. Separate digits of a given no. e.g. 351—1,5,3.
4. Count how many digits are there in a given no. e.g. 351-3.
5. Summation of digits of a given no. e.g. 351 = 1+5+3 = 9
6. Reverse digits of a given no. e.g. 351—153.
7. Check whether a given no. is palindrome no or not. e.g. 12321.
8. Find out whether a given no. is armstrong no. or not. If sum of cube of digits = no. itself then it is the armstrong no. e.g. 153.
9. Find out factors of a given no. e.g. 6-1,2,3,6
10. Find out whether a given no. is perfect no. or not.

If sum of factors upto half of a given no.=no. itself then no. is perfect no. e.g. 6=1+2+3=6

1. Check whether a given no. is prime no. or not e.g. 7,11,13,17.
2. Find out all prime nos. between 1and 500.
3. Find out summation of prime nos. between 1 & 500.
4. Find out how many prime nos. are there between 1 and 500.
5. Check whether a given no. is automorphic no. or not.

Automorphic nos. are the nos. whose last digit(s) of square of any no=no itself. e.g. 5-25,6-36,25-625

1. Print fibonacci series. 1,1,2,3,5,8,…upto n nos.
2. Print the following series. (loop within loop)
3. 1 1 (B) 1 1 (C) 1 1 1 (D) 5 1

1 2 2 1 1 1 2 5 2

1 3 2 2 1 2 1 4 1

2 1 3 1 1 2 2 4 2

2 2 3 2 2 1 1 3 1

2 3 3 3 2 1 2 3 2

2 2 1

2 2 2

1. Print the following series (Parallel loop)
2. 1 5 (B) 1 5

2 4 1 5

3 3 2 4

4 2 2 4

5 1 3 3

3 3

**Unit 2**

**WRITE C PROGRAMS FOR THE FOLLOWING. (ARRAYS)**

1. Accept 5 values and print them later on.
2. Accept 10 values and print 4th, 7th and 9th value.
3. Accept 5 values and sort the array in either ascending or descending order.

Find out different techniques for sorting the array and also find out which one is the best technique.

1. Print minimum no. of notes required. Notes given are 1, 2, 5, 10, 20, 50, 100, 200 & 500. e.g if value is 1256, then answer is 2 notes of 500 Rs., 1 note of 200 Rs., 1 note of 50 Rs., 1 note of 5 Rs., and 1 note of 1 Rupee.
2. Add two 2D array of same size and store the result in the 3rd one.
3. Multiply two 2D array and store the result in the 3rd 2D-array.
4. Obtain transpose of a 4x4 matrix. The transpose of a matrix is obtained by exchanging the elements of each row with the elements of the corresponding column.
5. Copy one array of 5 elements to another array of 10 elements skipping one element.
6. Reverse the array of maximum 5 elements.
7. Find out frequency of each number in the array (having 10 elements).
8. Shift all nos. by given n positions within an array of 10 elements either on left side or right side. Pad the remaining position with 0.
9. Insert a new number at the beginning of the array.
10. Insert a new number at a particular position within an array.
11. Insert a new number at a last position within an array.
12. Delete a value from the first position of the array.
13. Delete a value from a particular position within an array.
14. Delete a value from the last position.
15. Delete a value from the array.
16. Search a value within an array.

**/\* Various Array Operations.**

Assumption:No data loss

variable index & array a will be available throughout the program.

------------------------------------------------

1. Getvalues. (Fill the whole array with values.)

2. Traverse (Display).

3. Insert -> at beginning.

-> in between.

-> at end.

4. Delete by Position.

-> from beginning position.

-> from in between position.

-> from last position.

5. Delete by Value.

6. Search -> By Value.

-> By Position.

7. Reverse the Array.

8. Sorting the Array.

9. Frequency of the values.

-----------------------------------------------

A -> Array.

LB -> Lower Boundary. (0)

UB -> Upper Boundary. (size - 1)

-----------------------------------------------

index -> -1 // Index of the Array.

-----------------------------------------------

Getvalues(A,UB)

{

1 i = 0;index=-1;

2 Repeat Step No. 3 to 8 while ( i <= UB)

3 {

4 Accept Val

5 A[i] = val

6 i++

7 index++

8 }

}

-----------------------------------------------

traverse(A)

{

1 i = 0;

2 Repeat Step No. 3 to 7 Until ( i > index)

3 {

4 write A[i]

6 i++

7 }

}

------------------------------------------------

traverse2(A,LB,UB) // Display the content of array from a particular

// position to a Particular Position.

{

1 i = LB;

2 Repeat Step No. 3 to 7 Until ( i > UB)

3 {

4 write A[i]

6 i++

7 }

}

------------------------------------------------

Insertatbeginning(A,UB,val)

{

1 if ( index >= UB)

2 write "Can't Insert as Whole array is filled."

3 else

4 if ( index == -1) // Empty array.

5 {

6 index++

7 A[index] = val

8 }

9 else

10 {

11 // Array Partially filled and Partially empty.

12 // Shift all the elements to the right by 1 position.

13 i = index

14 Repeat step no. 15 to 18 Until ( i < 0)

15 {

16 A[i+1] = A[i]

17 i--

18 }

19 // Shifting completed... Now insert at the first position.

20 index++;

21 A[0] = val

}

------------------------------------------------

insertinbetween(A,UB,pos,val)

{

1 if (index == UB)

2 {

3 // Houseful

4 write "Can't Insert as Whole array is filled."

5 }

6 else

7 if (pos >= 0 and pos <= index )

8 {

9 // Allow value to be inserted.

10 i = index

11 Repeat step no. 12 to 15 Until ( i < pos)

12 {

13 A[i+1] = A[i]

14 i--

15 }

16 A[pos] = val

17 index++

18 }

19 else

20 write "Position is Out of range of values already there."

}

------------------------------------------------

insertatend(A,UB,val)

{

1 if (index == UB)

2 {

3 // Houseful

4 write "Can't Insert as Whole array is filled."

5 }

6 else

7 {

8 index++

9 A[index] = val

10 }

}

------------------------------------------------

// Delete by position --> from beginning.

deletefromstart(A)

{

1 if (index == -1)

2 write "Can't delete as Array is already empty."

3 else

4 {

5 tmp = A[0]

6 // shift all elements to left.

7 for(i=1;i<=index;i++)

8 A[i-1] = a[i]

9 write tmp," is deleted."

10 index-- // As we have deleted a[0] and shifted all elements to

11 // its left, we have to reduce index by 1.

12 }

}

-----------------------------------------------

// Delete by position --> from inbetween.

deletefrominbetweenpos(A,pos)

{

1 if (index == -1)

2 write "Can't delete as Array is already empty."

3 else

4 if (pos > index)

5 write "Can't find the position from where to delete a value."

6 else

7 if (pos < 0 )

8 write "Invalid Position."

9 else

10 {

11 tmp = A[pos]

12 i = pos+1

13 repeat step no. 14 to 15 while (i <= index)

14 a[i-1] = a[i]

15 i++

16 index--

17 write tmp, "is deleted"

18 }

}

-----------------------------------------------

// Delete by position --> from last

deletefromlast(A)

{

1 if (index == -1)

2 write "Can't delete as Array is already empty."

3 else

4 {

5 write A[index], "is deleted."

6 index--

7 }

}

// The people speak the word, the meaning is pole apart.

-------------------------------------------------

deletebyvalue(A,val) // for single value.

{

1 flag = 'n' // Assuming that the value is not found.

2 for (i = 0 ; i <= index ; i++)

3 {

4 if (A[i] == val)

5 {

6 // means value found.

7 flag = 'y'

8 for (j=i+1;j<= index ; j++)

9 {

10 A[j-1] = A[j]

11 }

12 index--

13 write val, " is deleted from position ", i

14 go to step no. 21

15 }

16 }

17 if (flag == 'n')

18 {

19 write val, "Not Found..."

20 }

21

}

// reverse the Array

reversearray(A)

{

1 i=0

2 j=index

3 Repeat step no.4 to 10 while(i<j)

4 {

5 tmp = a[i]

6 a[i] = a[j]

7 a[j] = tmp

8 i++

9 j--

10 }

}

\*/

#include <stdio.h>

#include <conio.h>

#include <stdlib.h>

#define MAX 5 // maximum 10 for correct graphical presentation.

#define AND &&

int index = -1;

int a[MAX] = {0};

void traverse();

void displaybox();

void main()

{

int ch = 0;

int ub = MAX-1;

int val,pos;

void Getvalues(int);

void Insertatbeginning(int ,int );

void insertinbetween(int ,int ,int );

void insertatend(int ,int );

void deletebyvalue(int );

void deletefromstart(void);

void deletefrominbetweenpos(int);

void deletefromlast(void);

void searchval(int );

void searchpos(int ,int );

void reversearray(void);

void sortarray(void);

void freq(void);

while (1)

{

clrscr();

printf("Main Menu\n");

printf("1. Get Value (Fill the whole array)\n");

printf("2. Traverse (Display).\n");

printf("3. Insert -> at beginning.\n");

printf("4. Insert -> in between.\n");

printf("5. Insert -> at end.\n");

printf("6. Delete by Position.\n");

printf(" 7. Delete value from 1st position.\n");

printf(" 8. Delete value from a particular position.\n");

printf(" 9. Delete value from last position.\n");

printf("10.Delete by Value.\n");

printf("11.Search -> By Value.\n");

printf("12.Search -> By Position.\n");

printf("13.Reverse the Array.\n");

printf("14.Sorting the Array.\n");

printf("15.Frequency of the values.\n");

printf("16.Exit\n");

displaybox();

do

{

gotoxy(1,18);

printf("Enter your choice [1...16] ");

scanf("%d",&ch);

} while (ch > 16 || ch < 1);

switch(ch)

{

case 1: Getvalues(ub);

break;

case 2: traverse();

break;

case 3: printf("Enter the value to be inserted. ");

scanf("%d",&val);

Insertatbeginning(ub,val);

break;

case 4: printf("Enter the value to be inserted. ");

scanf("%d",&val);

printf("Enter the position: ");

scanf("%d",&pos);

insertinbetween(ub,pos,val);

break;

case 5: printf("Enter the value to be inserted. ");

scanf("%d",&val);

insertatend(ub,val);

break;

case 7: deletefromstart();

break;

case 8: printf("Enter the position: ");

scanf("%d",&pos);

deletefrominbetweenpos(pos);

break;

case 9: deletefromlast();

break;

case 10:printf("Enter the value to be deleted. ");

scanf("%d",&val);

deletebyvalue(val);

break;

case 11:printf("Enter the value to be searched. ");

scanf("%d",&val);

searchval(val);

break;

case 12:printf("Enter the position: ");

scanf("%d",&pos);

searchpos(pos,ub);

break;

case 13: reversearray();

break;

case 14:sortarray();

break;

case 15: freq();

break;

case 16: exit(0);

}

}

};

void Getvalues(int UB)

{

// int i = 0,val;

index = -1;

printf("Getvalues::Please enter %d values one by one.\n",MAX);

while (index < UB)

{

index++;

gotoxy(1,index+20);

printf("Enter a Number a[%d]: ",index);

scanf("%d",&a[index]);

// a[i] = val;

// i++;

displaybox();

}

gotoxy(1,index+21);

printf("Value of index = %d\n",index);

getch();

// traverse();

}

void traverse()

{

int i = 0;

if (index == -1 )

{

gotoxy(50,23);

printf("traverse::Empty Array.\n");

}

else

{

printf("traverse::The values of the Array are : ");

while (i <= index)

{

printf("%d ",a[i]);

i++;

}

printf("\nValue of index = %d\n",index);

}

getch();

}

void Insertatbeginning(int UB,int val)

{

int i;

if ( index >= UB)

printf("Insertatbeginning::Can't Insert as Whole array is filled.");

else

if ( index == -1) // Empty array.

{

index++;

a[index] = val;

printf("Insertatbeginning::%d is placed at a[0] ",val);

}

else

{

// Array Partially filled and Partially empty.

// Shift all the elements to the right by 1 position.

i = index;

while ( i >= 0)

{

a[i+1] = a[i];

i--;

}

// Shifting completed... Now insert at the first position.

a[0] = val;

index++;

printf("Insertatbeginning::%d is placed at a[0] ",val);

}

printf("\nInsertatbeginning::Value of index = %d\n",index);

// traverse();

displaybox();

getch();

}

void insertinbetween(int UB,int pos,int val)

{

int i;

if (index == UB)

{

// Houseful

printf("inserinbetween::Can't Insert as Whole array is filled.\n");

}

else

if (pos >= 0 AND pos <= index )

{

// Allow value to be inserted.

i = index;

while ( i >= pos)

{

a[i+1] = a[i];

i--;

};

a[pos] = val;

printf("insertinbetween::%d is placed at a[%d]\n",val,pos);

index++;

}

else

printf("insertinbetween::Position is Out of range of values already there.\n");

// traverse();

displaybox();

getch();

}

void insertatend(int UB,int val)

{

if (index == UB)

{

// Houseful

printf("insertatend::Can't Insert as Whole array is filled.\n");

}

else

{

index++;

a[index] = val;

printf("insertatend::%d is placed at a[%d]\n",val,index);

}

// traverse();

displaybox();

getch();

}

// Delete by position --> from beginning.

void deletefromstart()

{

if (index == -1)

printf("deletefromstart::Can't delete as Array is already empty.\n");

else

{

int i;

int tmp = a[0];

// shift all elements to left.

for(i=1;i<=index;i++)

a[i-1] = a[i];

printf("deletefromstart::%d is deleted.\n",tmp);

index--; // As we have deleted a[0] and shifted all elements to

// its left, we have to reduce index by 1.

}

// traverse();

displaybox();

getch();

}

void deletebyvalue(int val) // for all values.

{

char flag = 'n'; // Assuming that the value is not found.

int i,j;

traverse();

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

{

if (a[i] == val)

{

// means value found.

flag = 'y';

for (j=i+1;j<= index ; j++)

a[j-1] = a[j];

gotoxy(1,22);

printf("%d is deleted from a[%d] position\n", val,i);

index--;

i--;

displaybox();

getch();

// break;

}

}

gotoxy(1,22);

if (flag == 'n')

printf ("%d Not Found...\n",val);

traverse();

}

void deletefrominbetweenpos(int pos)

{

if (index == -1)

printf("Can't delete as Array is already empty.\n");

else

if (pos > index)

printf("Can't find the position from where to delete a value.\n");

else

if (pos < 0 )

printf("Invalid Position.\n");

else

{

int tmp = a[pos];

int i = pos+1;

while (i <= index)

{

a[i-1] = a[i];

i++;

}

index--;

printf("%d is deleted.",tmp);

displaybox();

}

getch();

}

// Delete by position --> from last

void deletefromlast()

{

if (index == -1)

printf("Can't delete as Array is already empty.\n");

else

{

printf("%d is deleted.\n",a[index]);

index--;

}

// traverse();

displaybox();

getch();

}

// The people speak the word, the meaning is pole apart.

void searchval(int val)

{

int i;

char flag = 'n';

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

{

if (a[i] == val) // value found.

{

flag = 'y';

printf("%d is found at a[%d].\n",val,i);

}

}

if (flag == 'n')

printf("%d is not found.\n",val);

getch();

}

void searchpos(int pos,int ub)

{

if (pos < 0 || pos > ub)

printf("Invalid Position.\n");

else

if (pos > index)

printf("Array is not filled till this position.\n");

else

printf("%d is found at a[%d].\n",a[pos],pos);

getch();

}

void reversearray()

{

int i=0,j=index,tmp;

while(i<j)

{

tmp = a[i];

a[i] = a[j];

a[j] = tmp;

i++;

j--;

displaybox();

getch();

gotoxy(1,20);

}

}

void sortarray()

{

int i,j,tmp;

printf("Before Sorting::");

traverse();

for(i=0;i<=index-1;i++)

{

for (j=i+1;j<=index;j++)

{

if (a[i] > a[j])

{

tmp = a[i];

a[i] = a[j];

a[j] = tmp;

displaybox();

getch();

}

}

}

gotoxy(1,21);

printf("After Sorting::");

traverse();

}

void freq()

{

int b[MAX]={0},bfreq[MAX]={0},bindex=-1,i,j;

char valfound;

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

{

valfound = 'n'; // Assumed that value not found in array b.

// so now try to find it in array b.

for(j=0;j<=bindex;j++)

{

if (a[i] == b[j])

{

// means found in array b. so add 1 to its frequency.

bfreq[j] = bfreq[j] + 1;

valfound = 'y'; // reverse your assumption.

break;

}

}

if (valfound == 'n') // means value still not found. So add in b[].

{

bindex++;

b[bindex] = a[i];

bfreq[bindex] = 1;

}

}

// frequency of each value counted ... now print them one by one.

printf("Value\tFrequency\n");

for (j= 0; j<= bindex ; j++)

{

printf("%d\t%d\n",b[j],bfreq[j]);

}

getch();

}

void displaybox()

{

int r= 3 , c = 60,i;

// print value of index.

gotoxy(c+10,r-1);

printf("index");

gotoxy(c+10,r);

printf("%c",201);

printf("%c%c%c%c%c%c",205,205,205,205,205,205);

printf("%c",187);

gotoxy(c+10,r+1);

printf("%c",186);

printf("%6d%c",index,186);

gotoxy(c+10,r+2);

printf("%c",200);

printf("%c%c%c%c%c%c",205,205,205,205,205,205);

printf("%c",188);

// now display array.

gotoxy(c+3,r-1);

printf("A");

// print 1st line.

gotoxy(c,r);

printf("%c",201);

printf("%c%c%c%c%c%c",205,205,205,205,205,205);

printf("%c",187);

// print MAX-1 lines.

for(i = 0;i < MAX-1 ; i++)

{

r++;

gotoxy(c-2,r);

printf("%2d%c",i,186);

if ( i <= index)

printf("%6d%c",a[i],186);

else

printf(" %c",186);

r++;

gotoxy(c,r);

printf("%c",204);

printf("%c%c%c%c%c%c",205,205,205,205,205,205);

printf("%c",185);

};

// print last line.

r++;

gotoxy(c-2,r);

printf("%2d%c",i,186);

if ( i <= index)

printf("%6d%c",a[i],186);

else

printf(" %c",186);

r++;

gotoxy(c,r);

printf("%c",200);

printf("%c%c%c%c%c%c",205,205,205,205,205,205);

printf("%c",188);

// getch();

}

**// Implementation of Stack**

#include <stdio.h>

#include <conio.h>

#include <stdlib.h>

#define MAX 5

#define AND &&

#define UB (MAX-1)

int top = -1;

int a[MAX]={0};

void push(int );

void pop(void);

void traverse(void);

void peep(int );

void change(int,int);

void main()

{

int ch = 0;

int val,pos;

while (1)

{

clrscr();

printf("Main Menu - Stack\n");

printf("1. Push Value on the Stack.\n");

printf("2. Pop value from the Stack.\n");

printf("3. Traverse the Stack.\n");

printf("4. Peep in the Stack.\n");

printf("5. Change value in the Stack.\n");

printf("6. Exit.\n");

do

{

printf("Enter your choice [1...6] ");

scanf("%d",&ch);

} while (ch > 6 || ch < 1);

switch(ch)

{

case 1: printf("Enter the value to be pushed :");

scanf("%d",&val);

push(val);

break;

case 2: pop();

break;

case 3: traverse();

break;

case 4: if (top+1<1)

{

printf("Stack is empty.\n");

getch();

}

else

{

printf("Enter the position: [1..%d] : ",top+1);

scanf("%d",&pos);

peep(pos);

}

break;

case 5: printf("Enter the position: [1..%d] : ",top+1);

scanf("%d",&pos);

printf("Enter the value to be changed. ");

scanf("%d",&val);

change(pos,val);

break;

case 6: exit(0);

}

}

};

void change(int i,int val)

{

int pos = top - i + 1;

if (pos < 0)

printf("Stack is underflow on change.\n");

else

if (pos > top)

printf("Stack is overflow on change.\n");

else

{

printf("Old value : %d\n",a[pos]);

a[pos] = val;

printf("New value : %d\n",a[pos]);

}

getch();

}

void push(int val)

{

if (top == UB)

{

// Stack Houseful

printf("Stack is Full. (overflow)\n");

}

else

{

top++;

a[top] = val;

printf("%d is placed at [%d]\n",val,top);

getch();

}

}

void pop()

{

if ( top == -1)

printf("Stack is empty. (Underflow)\n");

else

{

printf("%d is popped.\n",a[top]);

top--;

}

getch();

}

void peep(int p)

{

int pos;

pos = top-p+1;

if (pos < 0 || pos > top)

printf("Invalid Position\n");

else

printf("%d is found at position %d\n",a[pos],p);

getch();

}

void traverse()

{

int i;

if (top == -1)

printf("Stack is empty.\n");

else

{

for(i=top;i>=0;i--)

{

printf("%d ",a[i]);

}

}

getch();

}

**// Implementation of Queue using array.**

#include <stdio.h>

#include <conio.h>

#include <stdlib.h>

#define MAX 10

int q[MAX] = {0};

int ub = MAX-1;

int f = -1, r = -1;

void traverseq(void);

void main()

{

void insertq(void);

void deleteq(void);

int ch;

do

{

clrscr();

// Display Main Menu.

printf("Q U E U E - M A I N M E N U\n");

printf("1.Insert value in Q.\n");

printf("2.Delete value from Q.\n");

printf("3.Traverse the Q.\n");

printf("4.Exit\n");

printf("Enter your choice...[1..4] ");

scanf("%d",&ch);

switch(ch)

{

case 1 : insertq();

break;

case 2 : deleteq();

break;

case 3 : traverseq();

break;

case 4 : exit(0);

default: printf("Invalid Choice. Please Re-enter.");

getch();

}

} while (1);

getch();

};

void insertq(void)

{

int val;

if (r == ub)

printf("Queue is Full.\n");

else

{

printf("Enter the value : ");

scanf("%d",&val);

r++;

q[r] = val;

if ( f == -1)

f=0; // as good as f++. Ultimately f will contain 0.

}

traverseq();

}

void deleteq(void)

{

// Check whether Q is already empty.

if (f == -1)

printf("Queue is Empty.\n");

else

{

// there will be at least one element in the Q.

printf("%d is deleted.\n",q[f]);

f++;

// Now check whether q has become empty or not.

if ( f > r)

f = r = -1;

}

traverseq();

}

void traverseq(void)

{

int i;

if ( f == -1)

printf("Queue is Empty...\n");

else

{

for (i=f;i<=r;i++)

printf("q[%d] = %d\t",i,q[i]);

printf("\n");

printf("Front = %d\nRear = %d\n",f,r);

}

getch();

}

**// Implementation of Circular Queue using array.**

#include <stdio.h>

#include <conio.h>

#include <stdlib.h>

#define MAX 3

int q[MAX] = {0};

int ub = MAX-1;

int f = -1, r = -1;

void traverseq(void);

void main()

{

void insertq(void);

void deleteq(void);

int ch;

do

{

clrscr();

// Display Main Menu.

printf("C I R C U L A R Q U E U E - M A I N M E N U\n");

printf("1.Insert value in Q.\n");

printf("2.Delete value from Q.\n");

printf("3.Traverse the Q.\n");

printf("4.Exit\n");

printf("Enter your choice...[1..4] ");

scanf("%d",&ch);

switch(ch)

{

case 1 : insertq();

break;

case 2 : deleteq();

break;

case 3 : traverseq();

break;

case 4 : exit(0);

default: printf("Invalid Choice. Please Re-enter.");

getch();

}

} while (1);

getch();

};

void insertq(void)

{

int val;

if (r == ub && f == 0 || r+1 == f)

{

printf("Queue is Full.\n");

getch();

}

else

{

printf("Enter the value : ");

scanf("%d",&val);

r++;

if (r > ub)

r = 0;

q[r] = val;

if ( f == -1)

f=0; // as good as f++. Ultimately f will contain 0.

}

}

void deleteq(void)

{

// Check whether Q is already empty.

if (f == -1)

printf("Queue is Empty.\n");

else

{

// there will be at least one element in the Q.

printf("%d is deleted.\n",q[f]);

f++;

// Now check whether q has become empty or not.

if (( f > ub) && (r < f))

f = 0;

else

if ( f-1 == r)

f = r = -1;

}

getch();

}

void traverseq(void)

{

int i;

if ( f == -1)

printf("Queue is Empty...\n");

else

{

if (f <= r)

{

for (i=f;i<=r;i++)

printf("q[%d] = %d\t",i,q[i]);

}

else

{

for (i=f;i<=ub;i++)

printf("q[%d] = %d\t",i,q[i]);

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

printf("q[%d] = %d\t",i,q[i]);

}

printf("\n");

printf("Front = %d\tRear = %d\n",f,r);

}

getch();

}

**WRITE C PROGRAMS FOR THE FOLLOWING. (STRINGS (use built-in functions))**

1. Find out length of a string.
2. Convert a string to a lower case.
3. Convert a string to an upper case.
4. Convert a string to a toggle case. E.g. AbC 🡪 aBc
5. Copy one string to another one.
6. Compare two strings lexicographically and print which one is greater or lower or print both strings are same.
7. Reverse the string.
8. Check whether a string is a Palindrome String or not.
9. Concatenate one string at the end of another one.
10. Print characters of a string vertically.
11. Print reversed string vertically character by character.
12. Print frequency of each vowel in a given string.

**WRITE C PROGRAMS FOR THE FOLLOWING. (USING Pointers, Arrays, Strings and Arrays of pointers)**

1. Perform addition, subtraction and multiplication operations on two matrices.
2. Sort all the elements of a 4x4 matrix and store the result in a single-dimension array.
3. Print the largest and smallest numbers from a 3x3 matrix using pointer.
4. Accept and print later on three books names using array of pointers.
5. Write a program that takes a set of names of individuals and abbreviates the first, middle and other names except the last name by their first letter.

**Unit 3**

**WRITE C PROGRAMS FOR THE FOLLOWING. (FUNCTIONS AND RECURSIVE FUNCTIONS)**

1. Write a function power (a, b), to calculate the value of a raised to b.
2. Any year is entered through the keyboard. Write a function to determine whether the year is a leap year or not.
3. Write a recursive function to calculate factorial of a number.
4. Write a function to swap two integers using call by value. Show that the original values are not changed.
5. Write a program that uses a function to update both the maximum and minimum values in an array using call by reference.
6. Write a program to implement a calculator using separate functions for add, subtract, multiply, and divide.
7. All the above mentioned programs
8. All the programs of Loop using recursion.

Programs related to Structures and Union

**Struct1c.cpp**

#include <conio.h>

#include <stdlib.h>

#include <stdio.h>

**// Declaring a Structure globally so that it is available through the program.**

struct book

{

char title[20]; // Attribute of the structure book.

char author[20]; // Attribute of the structure book.

int edition; // Attribute of the structure book.

char isbn[14]; // Attribute of the structure book.

char publisher[20]; // Attribute of the structure book.

float price; // Attribute of the structure book.

};

int main()

{

clrscr();

printf("Structure Called book is already created.\n");

**// Creating objects/entities from the structure. b1 & b2 are known as objects/entities.**

book b1, b2;

printf("Objects b1 & b2 of structure book are created.\n");

**// Creating objects and initialization simultaneously from the structure.**

book b3 = { "Let Us C","YK",4,"81-7656-940-2","BPB",180 };

**// Display the contents of the b3 object.**

printf("The content of object b3 is ... \n");

printf("Title : %s\n",b3.title);

printf("Author : %s\n",b3.author);

printf("Edition : %d\n",b3.edition);

printf("ISBN : %s\n",b3.isbn);

printf("Publisher : %s\n",b3.publisher);

printf("Price : %.2fs\n",b3.price);

**// copying the content of one object of structure into another object.**

b1 = b3;

**// Shall we repeat all the above statement for b1 object?**

**// No, we should find another way to do the job easily.**

**// .......................Thinking...............**

**//...........................ok..................**

**// Now pass the whole object into a function as an argument.**

**// Declaring a function having structure as an argument.**

void display(struct book);

**// call display function.**

printf("The content of object b1 is ... \n");

display(b1);

**// Accept the information from the user and store it into b2 object.**

printf("Enter information for b2 object.\n");

printf("Enter a Book Name : ");

gets(b2.title);

printf("Enter Author Name : ");

gets(b2.author);

printf("Enter Edition No. : ");

scanf("%d",&b2.edition);

printf("Enter ISBN : ");

flushall();

gets(b2.isbn);

printf("Enter Name of Publisher : ");

gets(b2.publisher);

printf("Enter Price of the Book : ");

flushall();

scanf("%f",&b2.price);

printf("The content of object b2 is ... \n");

display(b2);

**// Passing individual element of the structure.**

void display\_price(float,char \*);

display\_price(b1.price,"Book 1");

display\_price(b2.price,"Book 2");

display\_price(b3.price,"Book 3");

getch();

};

**// This function displays all the elements of the structure book.**

void display(struct book b)

{

**// Display the contents of the b object.**

printf("Title : %s\n",b.title );

printf("Author : %s\n",b.author );

printf("Edition : %d\n",b.edition );

printf("ISBN : %s\n",b.isbn );

printf("Publisher : %s\n",b.publisher );

printf("Price : %.2f\n",b.price );

}

**// This function displays price of the book.**

void display\_price(float cost,char \* message)

{

printf("%s\'s cost is %.2f\n",message, cost );

}

**Struct2c.cpp**

#include <conio.h>

#include <stdlib.h>

#include <stdio.h>

**// Declaring a Structure globally so that it is available throught the program.**

struct book

{

char title[20]; // Attribute of the structure book.

char author[20]; // Attribute of the structure book.

int edition; // Attribute of the structure book.

char isbn[14]; // Attribute of the structure book.

char publisher[20]; // Attribute of the structure book.

float price; // Attribute of the structure book.

};

int main()

{

// clrscr();

book b1; // creating a single object.

book b[100]; // Creating arrays of structure.

printf("Size of object b1 is %d\n",sizeof(b1));

printf("Size of array b is %d\n",sizeof(b));

**// Memory Addresses of different attributes of the book object.**

printf("Memory Address and size of different attributes of b1 object:\n");

printf("Title : %u (%d)\n",&b1.title, sizeof(b1.title));

printf("Author : %u (%d)\n", &b1.author, sizeof(b1.author));

printf("Edition : %u (%d)\n", &b1.edition, sizeof(b1.edition));

printf("ISBN : %u (%d)\n", &b1.isbn, sizeof(b1.isbn));

printf("Publisher : %u (%d)\n", &b1.publisher, sizeof(b1.publisher));

printf("Price : %u (%d)\n", &b1.price, sizeof(b1.price));

int i=0;

**for (;i<=2;i++)**

{

// Accept the information from the user and store it into array.

printf("Enter information for b[%d] object.\n",i);

printf("Enter a Book Name : ");

gets(b[i].title);

printf("Enter Author Name : ");

gets(b[i].author);

printf("Enter Edition No. : ");

scanf("%d",&b[i].edition);

printf("Enter ISBN : ");

flushall();

gets(b[i].isbn);

printf("Enter Name of Publisher : ");

gets(b[i].publisher);

printf("Enter Price of the Book : ");

flushall();

scanf("%f",&b[i].price);

flushall();

}

**// Display the contents of all objects using pointer.**

void display (struct book \*); // function prototype

for(i=0;i<=2;i++)

{

display(&b[i]);

}

getch();

};

**// This function displays all the elements of the structure book using pointer.**

void display(struct book \* pb)

{

// Display the contents of the b object.

printf("Title : %s\n", pb->title);

printf("Author : %s\n", pb->author);

printf("Edition : %d\n", pb->edition);

printf("ISBN : %s\n", pb->isbn);

printf("Publisher : %s\n", pb->publisher);

printf("Price : %.2f\n", pb->price);

}

**Struct3c.cpp**

**// this program makes u understand the concept of nested structure.**

#include <conio.h>

#include <stdlib.h>

#include <stdio.h>

struct address

{

char add1[50];

char add2[50];

char city[50];

long int pin;

};

struct studentinfo

{

int rlno;

char name[50];

address p;

address t;

};

int main()

{

//clrscr();

studentinfo s1;

**// Get the details of the student from the user.**

printf("Enter Roll No: ");

scanf("%d",&s1.rlno);

flushall();

printf("Enter Name : ");

gets(s1.name);

printf("Enter Permenant Address: \n");

printf("Line 1 :");

gets(s1.p.add1);

printf("Line 2 :");

gets(s1.p.add2);

printf("City :");

gets(s1.p.city);

printf("Pin :");

scanf("%ld",&s1.p.pin);

flushall();

printf("Enter Temporary Address: \n");

printf("Line 1 :");

gets(s1.t.add1);

printf("Line 2 :");

gets(s1.t.add2);

printf("City :");

gets(s1.t.city);

printf("Pin :");

flushall();

scanf("%ld",&s1.t.pin);

printf("\nThe content that you entered just right now is ... \n");

printf("Roll No.: ");

printf("%d\n",s1.rlno);

printf("Name : ");

printf("%s\n",s1.name);

printf("Permenant Address is ... \n");

printf("%s\n",s1.p.add1);

printf("%s\n",s1.p.add2);

printf("%s - %ld\n",s1.p.city, s1.p.pin) ;

printf("Temporary Address is ... \n");

printf("%s\n",s1.t.add1);

printf("%s\n",s1.t.add2);

printf("%s - %ld\n",s1.t.city, s1.t.pin) ;

getch();

}

Programs related to Structures and Union

1. Create a structure to specify data on students given below.  Roll number, Name, Course Name, Major and Minor Subjects.

Assume that there are 10 students.

a) The function should print name of all students.

b) Print the data of a student whose roll number is given.

1. Create a structure to specify data of customers in a bank. The data to be stored is: Account Number, Name, Balance in account. Assume maximum 10 customers in the bank.
   1. The function should print the account number and name of each customer with balance below Rs. 100.
   2. If a customer request for withdrawal or deposit, it is given in the form: Account Number, Amount, Code ( 1 for deposit, 0 for withdrawal)

The function should give a message, " The balance is insufficient for the specified withdrawal ".

1. Create a structure called student\_data containing following information only.

Roll No., name, marks of physics, maths and Chemistry, and total.

Create one instance from this structure.

Accept roll no., name, and marks of three subjects from the user and calculate total. Display all the information.

1. A record contains name of cricketer, his age, number of test matches that he has played and the average runs that he has scored in each test match.

Create an array of structure to hold records of 10 such cricketer and store some data in it. Write a program to arrange them in ascending order by average runs.

Unit 4

Programs related to File Handling

1. Read a file and count no.of lines, words, characters, spaces, in it.
2. Instead of characters, accept one line at a time from the user and write it into the file.(Hint: use gets() and fputs())
3. Read a content of the file line by line. (Hint: fgets())
4. Accept a record consisting of Roll No. and name from the user and store it in the file. The program should terminate when user is not willing to enter more records. The data stored in a file must be directly opened in MS-Excel.