Fundametal of Data Structures in c.

06/09/20.

Name : G. Mahalingam
class : E22 (ECE)
Roll.No : 13.

Part - A.

1. String handling function:
   * String length function - strlen().
   * string reverse function - str rev().
   * string copy function - str cpy().
   * string compare function - strcmp().

2. Types of C operators:
   * Arithmetic operators.
   * Assignment operators.
   * Increment Decrement operators
   * Relational operators
   * Logical operators.
   * Conditional operators.
   * Bitwise operators.
   * Special, comma, size of and address operators.

3. Advantages of using arrays:
   1. It is used to represent multiple data items of same type by using only single name.
   2. It is used to implement other structure like lists, stacks, queues, trees, graphs etc..
   3. 2D arrays are used to represent matrices.

4. Data types available in c:
   * char, unsigned char, signed char, int, unsigned int, signed int,

short int, unsigned short int, signed short int, long int, long long int, signed long int, unsigned long long int, float, double, long double, void.

5)

while loop.	do while loop.
Entry controlled loop	Exit controlled loop.
The condition is tested if true, the loop will be executed.	Loop will be executed once and then the condition is tested.
If the condition is false the loop will not get executed.	The loop will get executed once even if the condition is true.

6) Structure of C program:

```
main {}
  {
   Local declaration          }
   program statements         } Body of the
   calling using defined function } main () function.
  }
  user definar functions
   function1 }
   function2 } (option to user)
   function n }
```

7. Variable:
A variable is any characteristics, number or quantity that can be measured or counted. A variable may also be called data item.

Ex: Income is a varible that can vary between data units in a population.

8. Example for terrary operator:

```
int find maximum (int a, int b) {
      return (a>b) ? a : b ;
   }
```

9. These function are used to permit the transfer of information between the computer and the standard input /output device. The basic input/output function are get char, put char, puts, scanf and print f. The first two functions, get char, and put char are used to transfer single characters.

Part-B.

11).
a) Strings:

* String in C are represented as one dimensional character type array and each character with in this string represents one array.
* The string are always declared as the charactor array.
* Simply charactor array are called as String.

Declaration and initialization of Strings:

* A String is always declared ~~as~~ an array.
* The variable is any valid C variable name.
* There are several methods for declaring a variable.

method:-1
char variable - name [size];
Heresize represents the number of characters in the string.
Ex:- char name [25]:

method: 2
The syntax for declaring string using pointers is given below
char* var;
char* var = "dbcd";

Two ways to initialize
* At Compile time - at the time of declaring
* At runtime - at the time of executing at the time of executing.

At compile time initialized:
→ we can initialize the string at the time of declaration as follows.
* char name[5] = "CSE",
→ we can initialize the string during execution by using the scanf ().
* char name [5];
scanf("%s", name):

String length function - strlen().
* This function is used to find the length of the string.
Syntax:
n = strlen (string - variable):

Ex: charname [10] = "CSE",
int n;
n = strlen (s);
output:
Length of the string is 3

Scanned with CamScanner

String reverse function - Strrev()

* This function simply reverse the characters present in the String.

Syntax:-

```
strrev(string_variable);
```

Ex:-
```
char name[10] = "CSE";
strrev(s));
```
Output:- The reverse string is: ESC

String Copy function - strcpy()

* Here the second string is copied to the first string

* only the first string changed but the second string remains as it is.

Syntax:- strcpy(target_string, source_string);

Ex:- char s1[10] = "cse", s2[10 = "ece";

Output:-
```
first String: cse
second string: ece
After string copy
first string is: ece
second string is: ece.
```

String compare function - strcmp()

Syntax:- st 3cmp (string1, string2);

Ex:-
```
char s1[10] = "cse', s2[10 = "ece";
n = strcmp(s1, s2);
if (n==0)
{
print ("\n The given strings same");
}
else
{ print ("\n The string are different");
}
```

output: The given strings are differ ent.

11. b) Operators:

They are Symbols that indicate an operation and operation to be per formed. operators are used to manipulate data in program.

operators types }
- → Arithmetic operators.
- → Assignment operator.
- → increment Decrement operator.
- → Relational operator.
- → logical operator.
- → conditional operator
- → Bitwise operator.
- → special comma, size of and address operators.

Assignment ~~Arithmetic~~ operators:

Assignment operators are used to Combine the '=' operator with one of the binary arithmetic operators or simply gives value to variables combining assignment and arithmetic operators will reduce the number of registers in the operation c =9.

operator	Example	Equivalent Statement	Results.
+=	c+ = 7	c = c+7	c =16
-=	c- = 8	c = c-8	c =1
*=	c* = 10	c = c* 10	c=90
/=	c/ = 5	c = c/5	c = 1
%=	c% = 5	c = c%5	c = 4.

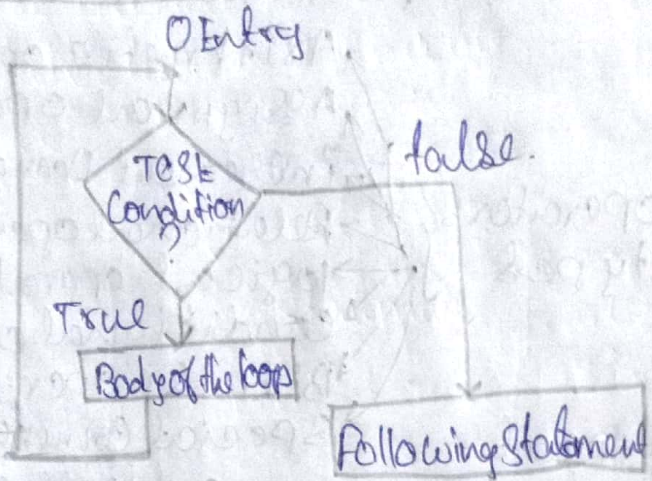
12) a). i) While loop:

* while loop is entry controlled, the condition is checked at the beginning of the loop.

* If the condition is false the loop will not be executed.

Syntax:-

```
while
{
  // Statement
}.
```



```
/* Sum of 1 to 10 numbers*/
# include <stdio.h>
int main()
{
  int i=1,sum=0;
  while(i<=10)
  {
          sum = sum+i;
            i=i+1
  }
 print("Total : %d", sum);
}
```

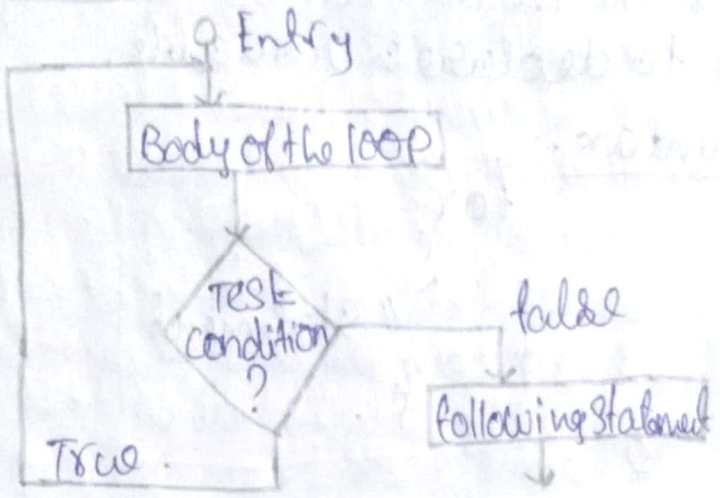
Output:- 55.

ii) Do:while loop:-

* If the loop repetition condition is true, the loop is repeated.
* otherwise, the loop is exited.

Syntax:

```
do
{
  // statment
}
while.
```



```
include <stdio.h>
int main()
{
 int count = 1;
 float x, sum = 0;
do
 {
   printf("x=");
   scanf("%f", &x);
   sum += x;
   ++ count
 } while (count <= 4);
 Print("Average = %f", (sum/4))
}
```

Output:

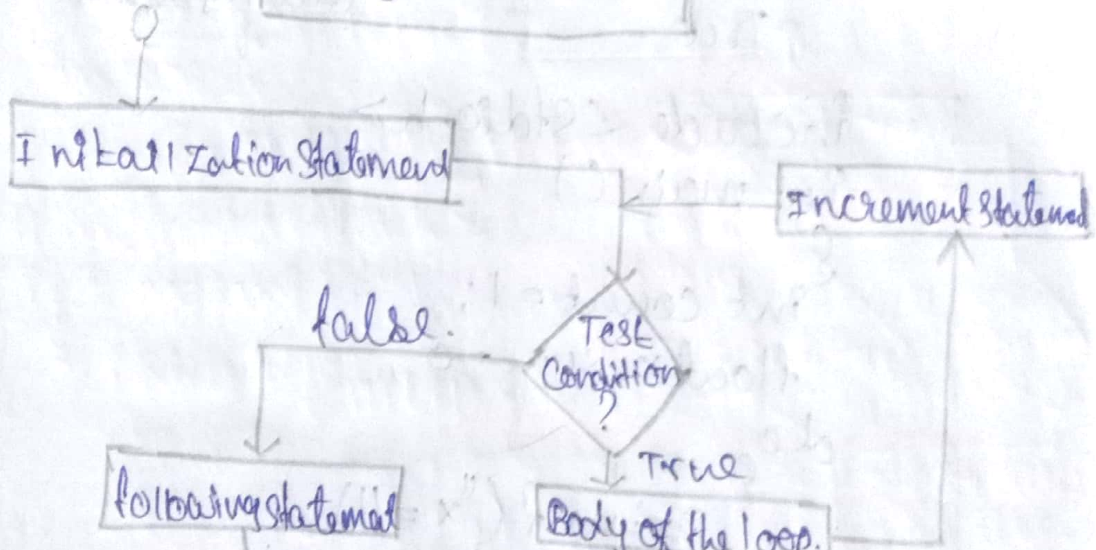
```
x = 6
x = 8
x = 43
x = 897

Average = 238.5
```

iii) For - loop :

* Initialization of the loop variable.
* Test of the loop repetition condition.
* change of the loop control variable.
* The initialization section can be used to declare a variable.

Syntax:

```
for
  {
    // statement
  }.
```



```
# include <stdio.h>
int main()
 {
   int n, i, factors = 0;
   printf("Enter a number:");
   scanf(" %d ", & n);
   for(i=1; i<=n; i++)
  {
    if((n%i)==0)++ factors;
  }
 if(factors==2)
Print("%.d is primenumber,"n);
else
  Print("%.d is not prime number,"n);
}
```

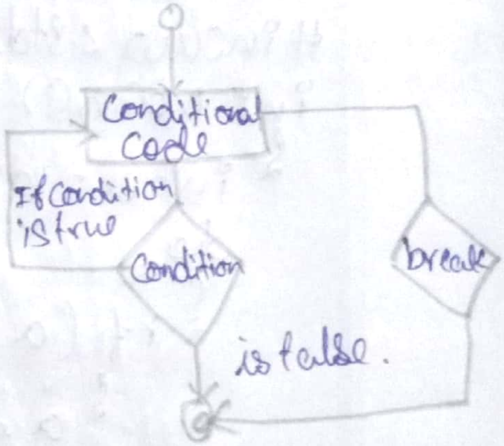
Output:
```
    Enter a number:7
    7 is Prime number.
```

12. b) Branching Statements:

i) Break Statement:

* when the break statement is encountered inside a loop, the loop is immediately terminated and program control resumes at the next statement following the loop.

* It can be used to terminate a case in the switch statement.



Syntax:
```
break;
```

```
#include
int main()
{ int a=10;
  while(a<20)
  {
   print("value of a: %.d\n", a);
   a++;
   if(a>15)
   { break; /*terminate the loop using break statement*/
   }
  }
  return();
}
```

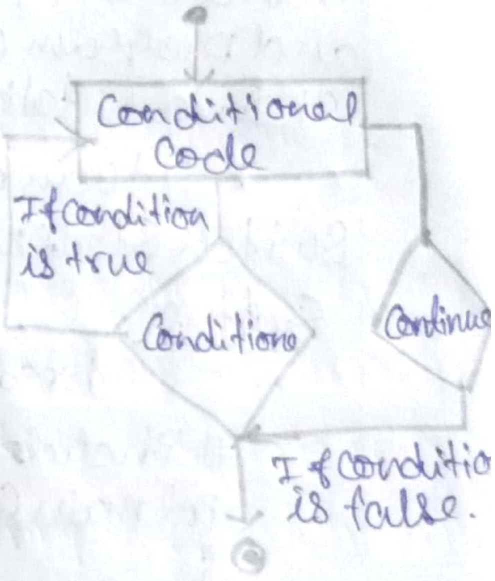
output:
```
value of a : 10
value of a : 11
value of a : 12
value of a : 13
value of a : 14
value of a : 15 .
```

ii) Continue Statement:

* The continue statement forces the next iteration of the loop to take place, skipping any code in between.
* for the while and do-while loops, continous statement causes the program control passes to the conditional test.

Syntax:

```
continue;
```



```
#includ <stdio.h>
int main()
{ int a=10;
  do
   { if (a==15)
      { a=a+1;
        continue;
      }
      printf ("value of a: %d\n", a);
      a++;
      } while (a<20);
   return0;
}
```

output:
```
value of a : 10
value of a : 11
value of a : 12
value of a : 13
```

```
Value of a : 14
Value of a : 15
value of a : 16
Value of a : 17
Value of a : 18.
```

13)

a). Two matrix C program.

```
#include <stdio.h>
int main()
{
int a[5][2] = {{0,0}, {1,2},{2,4},{3,6},{4,8}};
                    // an array with 5 rows and 2
                                        columns.
 int i,j;
 for(i=0; i<5; i++)
 { for(j=0; j<2; j++)
     {
       Printf("a[%d] = %d\n", i,j, a[i][j]
                                        );
     }
 } return 0;
}
```

Output:

```
    a[0][0] : 0
    a[0][1] : 0
    a[1][0] : 1
    a[1][1] : 2
    a[2][0] : 2
    a[2][1] : 4
    a[3][0] : 3
    a[3][1] : 6
    a[4][0] : 4
    a[4][1] : 8
```

Scanned with CamScanner

13.

b) reverse string:-

String reverse - strrev()

* The strrev() is used to ~~reserse the string~~

* This function simply reverse the characters ~~present in the string~~.

~~Syntax~~:

```
# include <stdio.h>
# include <conio.h>
# include <string.h>
void main()
{
  char str[50];
  clrscr();
  printf("\n\t Enter yourname");
  gets(str);
  printf("\n lowercase of string:%s",strlwr(str));
  printf("\n uppercase of string:%s",strupr(str));
  printf("\n Reverse of string: %s",strrev(str));
  printf("\n lengthof string:%d",strlen(str));
  getch();
}
```

output:

```
Enter your name : Mahalingam.
Lower case of string: mahalingam.
upper case of string: MAHALINGAM
Reverse case of string: MAGNILAHAM
Length of string : 10.
```

Scanned with CamScanner