Questions & Answers (Computer Programming using C Language - BCA 1st Semester)

Unit 1

1. Short Answer Questions

Q1. What is an algorithm?

Answer: An algorithm is a finite set of clear, step-by-step instructions used to solve a specific problem.

Q2. What is a flowchart?

Answer: A flowchart is a graphical representation of an algorithm using symbols to show the sequence of steps.

Q3. What is pseudocode?

Answer: Pseudocode is an informal way of writing program logic using simple English statements without following strict programming syntax.

Q4. What is a high-level language?

Answer: A high-level language is a programming language that is easy for humans to read and write, such as C, C++, Java, and Python.

Q5. What is a low-level language?

Answer: A low-level language is close to machine language and hardware-dependent, such as machine language and assembly language.

Q6. What is a compiler?

Answer: A compiler is a program that translates the entire high-level language program into machine code at once.

Q7. What is an interpreter?

Answer: An interpreter translates and executes a program line by line.

---

2. Fill in the Blanks

1. An __________ is a step-by-step procedure to solve a problem.
2. A __________ is used to represent an algorithm graphically.
3. __________ language is machine dependent.
4. C language is an example of a __________ level language.
5. A __________ converts the whole program at once.
6. An __________ converts the program line by line.

Answers:

1. Algorithm
2. Flowchart
3. Low-level
4. High-level
5. Compiler
6. Interpreter

---

3. Full Forms

1. CPU – Central Processing Unit
2. ALU – Arithmetic Logic Unit
3. HLL – High Level Language
4. LLL – Low Level Language
5. IDE – Integrated Development Environment

---

4. Definitions

Algorithm

A finite sequence of well-defined steps used to solve a problem.

Flowchart

A diagrammatic representation of an algorithm using standard symbols.

Pseudocode

A method of describing an algorithm using structured English statements.

Compiler

A software tool that converts a complete source code into machine code.

Interpreter

A software that translates and executes source code line by line.

---

5. Long Answer Questions / Explanations

Q1. Explain an Algorithm with its characteristics.

Answer:
An algorithm is a sequence of instructions to solve a problem.
Characteristics of an Algorithm:

1. Input – Accepts zero or more inputs
2. Output – Produces at least one output
3. Definiteness – Steps must be clear
4. Finiteness – Must terminate after finite steps
5. Effectiveness – Each step must be simple and executable

Q2. Explain Flowchart and its symbols.

Answer:
A flowchart visually represents the steps of an algorithm.

Common Flowchart Symbols:

- Oval – Start/Stop
- Parallelogram – Input/Output
- Rectangle – Process
- Diamond – Decision
- Flow lines – Direction of flow

Q3. Explain Pseudocode.

Answer:
Pseudocode helps programmers plan logic before writing actual code. It avoids syntax rules and focuses on logic.

Advantages:

- Easy to understand
- Language independent
- Simplifies coding

Q4. Differentiate between High-Level and Low-Level Language.

High-Level Language	Low-Level Language
Easy to understand	Difficult to understand
Machine independent	Machine dependent
Slower execution	Faster execution
Example: C, Java	Example: Assembly

Q5. Explain Compiler and Interpreter.

Answer:
A compiler translates the whole program at once and generates an executable file. An interpreter translates and executes code line by line.

---

6. Programming Examples (C Language)

Program 1: Algorithm to add two numbers

1. Start
2. Read two numbers A and B
3. Sum = A + B
4. Print Sum
5. Stop

Program 2: Pseudocode to find largest of two numbers

```
START
READ A, B
IF A > B
  PRINT A
ELSE
  PRINT B
END IF
STOP
```

Program 3: C Program to add two numbers

```c
#include <stdio.h>
#include <conio.h>
void main()
{
   int num1, num2, sum;
   clrscr();
   printf("Enter first number: ");
   scanf("%d", &num1);
   printf("Enter second number: ");
   scanf("%d", &num2);
   sum = num1 + num2;
   printf("The result is:%d", sum);
   getch();
}
```

Program 4: C Program to check greater number

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

```
int main() {
   int num1,num2 ;
   printf("Enter two Numbers: ");
   scanf("%d %d", &num);
   if (num1 > num2)
      printf("Number 1 is greater");
   else
      printf("Number 2 is greater");
   getch();
}
```

---

7. Very Short Questions

1. Name any high-level language.
   Answer: C

2. Name any low-level language.
   Answer: Assembly language

3. Which language is machine dependent?
   Answer: Low-level language

4. Which translator is faster in execution?
   Answer: Compiler

Unit 2

A. Very Short Answer Questions

1. **What is a C program?**
   A set of instructions written in C language to perform a task.

2. **What is the starting point of a C program?**
   `main()` function.

3. **What is a variable?**
   A memory location used to store data.

4. **What is a constant?**
   A value that does not change during program execution.

5. **What is an identifier?**
   A name given to variables, functions, or arrays.

B. Short Answer Questions

1. **Write the structure of a C program.**
   The structure of a C program includes documentation section, link section, definition section, global declaration section, `main()` function, and user-defined functions.

2. **What are keywords? Give examples.**
   Keywords are reserved words with predefined meanings in C. Examples: `int`, `float`, `if`, `return`.

3. **Define variable declaration.**
   Variable declaration specifies the data type and name of a variable before its use.

4. **What is assignment of value?**
   Assignment of value means storing a value in a variable using assignment operator (=).

5. **What are operators?**
   Operators are symbols used to perform operations on operands.

C. Fill in the Blanks

1. C program execution starts from the __________ function.
2. A variable must be __________ before use.
3. `=` is called __________ operator.

4. ++ is known as ___________ operator.

Answers:

1. main()
2. declared
3. assignment
4. increment

D. Full Forms

- IDE – Integrated Development Environment
- ASCII – American Standard Code for Information Interchange
- CPU – Central Processing Unit

E. Long Answer Questions

1. Explain the structure of a C program.

A C program consists of several sections:

1. **Documentation Section** – Comments describing the program
2. **Link Section** – Includes header files
3. **Definition Section** – Defines constants
4. **Global Declaration Section** – Global variables
5. **main() Function** – Program execution starts here
6. **User-defined Functions** – Additional functions

---

2. Explain basic data types in C.

Data Type	**Description**
int	Stores integers

float	Stores decimal values
char	Stores single character
double	Stores double precision values

---

3. Explain operators in C.

Operators are classified as:

- Arithmetic operators (+, -, *, /, %)
- Relational operators (<, >, <=, >=, ==, !=)
- Logical operators (&&, ||, !)
- Assignment operators (=, +=, -=)
- Increment and Decrement operators (++ , --)
- Conditional operator (?:)
- Bitwise operators (&, |, ^, ~, <<, >>)
- Special operators (sizeof, &, *)

---

4. Explain operator precedence and associativity.

Operator precedence determines the order of execution of operators. Associativity decides the direction of execution when operators have the same precedence.

---

5. Explain type conversion in C.

Implicit Type Conversion: Automatically done by compiler.

Explicit Type Conversion (Type Casting):

```
float b = (float)5;
```

F. Programming Questions

1. Write a C program to add two numbers.

```
#include <stdio.h>
#include <conio.h>
void main()
{
   int num1, num2, sum;
   clrscr();
   printf("Enter first number: ");
   scanf("%d", &num1);
   printf("Enter second number: ");
   scanf("%d", &num2);
   sum = num1 + num2;
   printf("The result is:%d", sum);
   getch();
}
```

---

2. Write a C program using conditional operator.

```
#include <stdio.h>
#include <conio.h>
void main() {
   int a = 10, b = 20;
   int max = (a > b) ? a : b;
   printf("Maximum = %d", max);
   getch();
}
```

---

3. Write a C program to demonstrate increment and decrement operators.

```
#include <stdio.h>
#include <conio.h>
void main()
{
   int a = 5, b = 10;
   clrscr();
   printf("Initial value of a = %d\n", a);
   printf("Post-increment a++ = %d\n", a++);
   printf("After post-increment, a = %d\n\n", a);
   printf("Pre-increment ++a = %d\n", ++a);
```

```
    printf("After pre-increment, a = %d\n\n", a);
    /* Decrement Operators */
    printf("Initial value of b = %d\n", b);
    printf("Post-decrement b-- = %d\n", b--);
    printf("After post-decrement, b = %d\n\n", b);
    printf("Pre-decrement --b = %d\n", --b);
    printf("After pre-decrement, b = %d\n", b);
    getch();
}
```

Unit 3

A. Very Short Answer Questions

1. **What is an if statement?**
   An if statement is a decision-making statement used to execute a block of code when a condition is true.

2. **What is an if–else statement?**
   It executes one block of statements if the condition is true, otherwise another block.

3. **What is a nested if–else statement?**
   An if–else statement inside another if–else statement is called nested if–else.

4. **What is a switch statement?**
   A switch statement is used to select one option from multiple choices.

5. **What is a loop?**
   A loop is used to repeat a set of statements.

6. **What is break statement?**
   The break statement is used to terminate a loop or switch statement.

7. **What is continue statement?**
   The continue statement skips the current iteration of a loop.

B. Short Answer Questions

1. **Explain if–else statement.**
   The if–else statement checks a condition. If the condition is true, the if block executes; otherwise, the else block executes.

2. **What is nested if–else?**
   Nested if–else is used to check multiple conditions by placing one if–else inside another.

3. **Explain switch–case statement.**
   The switch–case statement executes one case depending on the value of an expression.

4. **What is goto statement?**
   The goto statement transfers control to a labeled statement within the same function.

5. **Difference between while loop and do–while loop.**
   In while loop condition is checked first, whereas do–while loop executes at least once.

C. Fill in the Blanks

1. The ___________ statement is used for multiple selection.
2. The ___________ loop executes at least once.
3. The ___________ statement is used to exit from a loop.
4. The ___________ statement skips the current iteration.

Answers:

1. switch,
2. do–while,
3. break,
4. continue

D. Long Answer Questions

1. Explain if, if–else and nested if–else statements.

- **if statement:** Executes statements when condition is true.
- **if–else statement:** Executes one block if condition is true, otherwise another block.
- **Nested if–else:** Used to test multiple conditions.

---

2. Explain switch–case statement with syntax.

The switch statement allows multi-way branching.

Syntax:

```
switch(expression)
{
  case value1:
    statements;
    break;
  case value2:
    statements;
    break;
  default:
    statements;
}
```

---

3. Explain looping statements.

- **for loop:** Used when number of iterations is known.
- **while loop:** Used when condition is checked before execution.
- **do–while loop:** Executes at least once.

---

4. Explain break and continue statements.

- **break:** Terminates loop or switch statement.
- **continue:** Skips remaining statements of current iteration.

E. Programming Questions (Turbo C Format)

1. Program to check whether a number is even or odd (if–else)

```
#include <stdio.h>
#include <conio.h>
void main()
{
   int num, rem;
   clrscr();
   printf("Enter a number: ");
   scanf("%d", &num);
   rem = num % 2;
   if (rem==0)
  {
   printf("The number is even");
  }
   else
  {
   printf("The number is odd");
  }
   getch();
}
```

---

2. Program to find the largest of three numbers (nested if)

```
#include <stdio.h>
#include <conio.h>
void main()
{
   int a, b, c;
   clrscr();
   scanf("%d%d%d", &a, &b, &c);
   if(a > b)
   {
      if(a > c)
         printf("A is largest");
      else
         printf("C is largest");
   }
   else
   {
      if(b > c)
         printf("B is largest");
      else
         printf("C is largest");
   }
```

```
    getch();
}
```

---

3. Program using switch–case statement

```
#include <stdio.h>
#include <conio.h>
void main()
{
    int ch;
    clrscr();
    scanf("%d", &ch);
    switch(ch)
    {
        case 1: printf("One"); break;
        case 2: printf("Two"); break;
        default: printf("Invalid choice");
    }
    getch();
}
```

---

4. Program using for loop

```
#include <stdio.h>
#include <conio.h>
void main()
{
    int i;
    clrscr();
    for(i = 1; i <= 5; i++)
        printf("%d ", i);
    getch();
}
```

---

5. Program using while loop

```
#include <stdio.h>
#include <conio.h>
void main()
{
   int i = 1;
   clrscr();
   while(i <= 5)
   {
      printf("%d ", i);
      i++;
   }
   getch();
}
```

---

6. Program using do–while loop (Generate The Series of Odd number)

```
#include <stdio.h>
#include <conio.h>
void main()
{
   int fno=1, lno;
   clrscr();
   printf("Enter the last number: ");
   scanf("%d", &lno);
   do
  {
   printf("%d\t", fno);
   fno = fno + 2;
  }
   while(fno<=lno);
   getch();
}
```

---

7. Program demonstrating break and continue

```
#include <stdio.h>
#include <conio.h>
void main()
{
   int i;
   clrscr();
   for(i = 1; i <= 10; i++)
   {
      if(i == 5)
         continue;
      if(i == 8)
```

```
            break;
        printf("%d ", i);
    }
    getch();
}
```

Unit 4

A. Very Short Answer Questions

1. **What is an array?**
   An array is a collection of similar data types stored in contiguous memory locations.

2. **What is one-dimensional array?**
   An array that uses a single subscript is called one-dimensional array.

3. **What is a two-dimensional array?**
   An array with rows and columns is called a two-dimensional array.

4. **What is a string?**
   A string is a collection of characters terminated by null character `\0`.

5. **What is a pointer?**
   A pointer is a variable that stores the address of another variable.

B. Short Answer Questions

1. **How are array elements accessed?**
   Array elements are accessed using index numbers starting from zero.

2. **What is array initialization?**
   Assigning values to array elements at the time of declaration is called array initialization.

3. **What is multidimensional array?**
   An array having more than two dimensions is called a multidimensional array.

4. **Define string functions.**
   String functions are predefined functions used to perform operations on strings.

5. **What is pointer initialization?**
   Assigning the address of a variable to a pointer is called pointer initialization.

C. Fill in the Blanks

1. Array index starts from ___________.
2. A string ends with ___________ character.
3. `strlen()` function is used to find ___________ of a string.
4. Pointer variable stores ___________ of another variable.

Answers:

1. 0,
2. null (\0),
3. length,
4. address

D. Long Answer Questions

1. Explain arrays in C.

An array is used to store multiple values of the same data type under a single name. Each element is accessed using an index.

---

2. Explain two-dimensional arrays.

A two-dimensional array is stored in row and column format and is mainly used for matrix operations.

---

3. Explain strings and string functions.

Strings are arrays of characters terminated by `\0`.

Common string functions:

- `strlen()` – finds length
- `strcpy()` – copies string
- `strcat()` – concatenates string
- `strcmp()` – compares string

---

4. Explain pointers.

Pointers store memory addresses and help in efficient memory management and function calls.

E. Programming Questions (Turbo C Format)

1. Program to read and print elements of an array

```
#include <stdio.h>
#include <conio.h>
void main()
{
   int a[5], i;
   clrscr();
   for(i = 0; i < 5; i++)
      scanf("%d", &a[i]);
   for(i = 0; i < 5; i++)
      printf("%d ", a[i]);
   getch();
}
```

---

2. Program to add two matrices (2D array)

```
#include <stdio.h>
#include <conio.h>
void main()
{
   int a[2][2], b[2][2], c[2][2];
```

```
    int i, j;
    clrscr();
    for(i = 0; i < 2; i++)
        for(j = 0; j < 2; j++)
            scanf("%d", &a[i][j]);
    for(i = 0; i < 2; i++)
        for(j = 0; j < 2; j++)
            scanf("%d", &b[i][j]);
    for(i = 0; i < 2; i++)
        for(j = 0; j < 2; j++)
            c[i][j] = a[i][j] + b[i][j];
    for(i = 0; i < 2; i++)
    {
        for(j = 0; j < 2; j++)
            printf("%d ", c[i][j]);
        printf("\n");
    }
    getch();
}
```

---

3. Program to Find the length of the string using Library function

```
#include<stdio.h>
#include<string.h>
#include<conio.h>
void main()
{
    char ch[]="river";
    int length;
    length = strlen(ch);
    printf("The length of the string %s is %d\n", ch, length);
    getch();
}
```

---

4. Program demonstrating pointer usage

```
#include <stdio.h>
#include <conio.h>
void main()
{
    int a = 10;
    int *p;
    clrscr();
    p = &a;
    printf("Value = %d", *p);
    getch();
}
```

5. Program to access variable using address and pointer

```
#include <stdio.h>
#include <conio.h>
void main()
{
    int x = 5;
    int *p;
    clrscr();
    p = &x;
    printf("Value using variable = %d\n", x);
    printf("Value using pointer = %d", *p);
    getch();
}
```

Unit 5

A. Very Short Answer Questions

1. **What is a function?**
   A function is a block of code that performs a specific task.

2. **What is function declaration?**
   It tells the compiler about function name, return type, and parameters.

3. **What is function definition?**
   It contains the actual body of the function.

4. **What is call by value?**
   Passing copy of actual value to the function.

5. **What is call by reference?**
   Passing address of variables to the function.

6. **What is recursion?**
   A function calling itself.

7. **What is a structure?**
   A user-defined data type that stores different data types.

8. **What is a union?**
   A user-defined data type where all members share the same memory.

B. Short Answer Questions

1. **What are function arguments?**
   Values passed to a function when it is called are called arguments.

2. **Difference between call by value and call by reference.**
   Call by value does not change original value, call by reference changes original value.

3. **What is return type of a function?**
   It specifies the type of value returned by a function.

4. **What is passing array to a function?**
   Sending an array as an argument to a function.

5. **Difference between structure and union.**
   Structure allocates separate memory, union shares memory.

C. Fill in the Blanks

1. A function must be ___________ before calling.
2. ___________ keyword is used to return value from function.
3. In call by reference, ___________ of variable is passed.

4. Structure members are accessed using __________ operator.

Answers:

1. declared,
2. return,
3. address,
4. dot (.)

D. Long Answer Questions

1. Explain functions in C.

Functions are reusable blocks of code used to perform specific tasks. They make programs modular and easy to understand.

---

2. Explain call by value and call by reference.

- **Call by value:** Copy of variable is passed. Changes do not reflect back.
- **Call by reference:** Address is passed. Changes reflect back.

---

3. Explain recursion with example.

Recursion is a process where a function calls itself until a base condition is met.

---

4. Explain structures.

Structures allow grouping of different data types under one name.

---

5. Explain union.

Union stores different data types in the same memory location but only one value at a time.

E. Programming Questions (Turbo C Format)

1. Program to find sum of two numbers using function

```
#include <stdio.h>
#include <conio.h>
int sum(int a, int b);
void main()
{
   int x, y, s;
   clrscr();
   scanf("%d%d", &x, &y);
   s = sum(x, y);
   printf("Sum = %d", s);
   getch();
}
int sum(int a, int b)
{
   return a + b;
}
```

---

2. Program to demonstrate call by value

```
#include <stdio.h>
#include <conio.h>
void change(int x);
void main()
{
   int a = 5;
   clrscr();
   change(a);
   printf("Value = %d", a);
   getch();
}
void change(int x)
{
   x = 10;
}
```

---

3. Program to demonstrate call by reference

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

```
#include <conio.h>
void change(int *x);
void main()
{
   int a = 5;
   clrscr();
   change(&a);
   printf("Value = %d", a);
   getch();
}
void change(int *x)
{
   *x = 10;
}
```

---

4. Program to find factorial using recursion

```
#include <stdio.h>
#include <conio.h>
// function prototype
int fact(int n);
void main()         // Turbo C accepts void main
{
   int num, result;
   clrscr();        // optional: clears the screen
   printf("Enter the number: ");
   scanf("%d", &num);

   result = fact(num);
   printf("The factorial is %d", result);

   getch();         // wait for key press before exit
  }
   int fact(int n)      // recursive factorial function
  {
   if (n <= 1)
      return 1;                // base case
   else
      return n * fact(n - 1);   // recursive call
}
```

---

5. Program to pass array to a function

```
#include <stdio.h>
#include <conio.h>
void display(int a[], int n);
void main()
{
   int a[5] = {1,2,3,4,5};
   clrscr();
   display(a, 5);
   getch();
}
void display(int a[], int n)
{
   int i;
   for(i = 0; i < n; i++)
      printf("%d ", a[i]);
}
```

---

6. Program using structure (Name Price and Pages of Book)

```
#include<stdio.h>
#include<conio.h>
void main()
{
   struct book
   {
      char name[20];
      int price;
      int pages;
   };
   struct book b[3];
   int i;
   clrscr();
   for(i = 0; i < 3; i++)
   {
      printf("Enter name, price and pages: ");
      scanf("%s %d %d", b[i].name, &b[i].price, &b[i].pages);
   }
   for(i = 0; i < 3; i++)
   {
      printf("The name, price and pages are: %s %d %d\n",
           b[i].name, b[i].price, b[i].pages);
```

```
    }
    getch();
}
```

---

7. Program using union

```
#include <stdio.h>
#include <conio.h>
union data
{
    int a;
    float b;
};
void main()
{
    union data d;
    clrscr();
    d.a = 10;
    printf("a = %d", d.a);
    getch();
}
```

Unit 6

A. Very Short Answer Questions

1. **What is a file?**
   A file is a collection of data stored permanently on secondary storage.

2. **What is file handling?**
   File handling is the process of reading and writing data to a file.

3. **Which function is used to open a file?**
   `fopen()`

4. **Which function is used to close a file?**
   `fclose()`

5. **What is a text file?**
   A file that stores data in character format.

6. **What is a binary file?**
   A file that stores data in binary format.

7. **What is random access file?**
   A file where data can be accessed from any position.

B. Short Answer Questions

1. **What is the use of fopen()?**
   It opens a file in specified mode.

2. **What are file opening modes?**
   They define how a file is opened (read, write, append).

3. **Difference between text file and binary file.**
   Text files store readable characters, binary files store data in binary form.

4. **What is fseek()?**
   It moves file pointer to a specified location.

5. **What is ftell()?**
   It returns the current position of file pointer.

C. Fill in the Blanks

1. `fopen()` returns a __________ pointer.
2. `fclose()` is used to __________ a file.
3. `fseek()` is used for __________ access.
4. Binary files use __________ mode.
5. EOF stands for _________.

Answers:

1. FILE
2. close
3. random
4. binary
5. End Of File

D. Long Answer Questions

1. Explain file handling in C.

File handling allows storing data permanently. It uses functions like fopen(), fclose(), fprintf(), fscanf(), fread(), fwrite().

---

2. Explain text file and binary file.

- **Text File:** Stores data as characters, easy to read.
- **Binary File:** Stores data in binary form, faster and secure.

---

3. Explain random access to files.

Random access allows direct access to data using functions like fseek(), ftell(), rewind().

E. Programming Questions (Turbo C Format)

1. Program to write data into a file

```
#include <stdio.h>
#include <conio.h>
void main()
{
   FILE *fp;
   clrscr();
   fp = fopen("data.txt", "w");
   fprintf(fp, "Welcome to File Handling");
```

```
    fclose(fp);
    getch();
}
```

---

2. Program to read data from a file

```
#include <stdio.h>
#include <conio.h>
void main()
{
    FILE *fp;
    char ch;
    clrscr();
    fp = fopen("data.txt", "r");
    while((ch = fgetc(fp)) != EOF)
        printf("%c", ch);
    fclose(fp);
    getch();
}
```

---

3. Program to write data to a binary file

```
#include <stdio.h>
#include <conio.h>
void main()
{
    FILE *fp;
    int n = 10;
    clrscr();
    fp = fopen("num.dat", "wb");
    fwrite(&n, sizeof(n), 1, fp);
    fclose(fp);
    getch();
}
```

---

4. Program to read data from a binary file

```
#include <stdio.h>
#include <conio.h>
void main()
```

```
{
   FILE *fp;
   int n;
   clrscr();
   fp = fopen("num.dat", "rb");
   fread(&n, sizeof(n), 1, fp);
   printf("Number = %d", n);
   fclose(fp);
   getch();
}
```

---

5. Program to demonstrate random access using fseek()

```
#include <stdio.h>
#include <conio.h>
void main()
{
   FILE *fp;
   clrscr();
   fp = fopen("data.txt", "r");
   fseek(fp, 5, SEEK_SET);
   printf("%c", fgetc(fp));
   fclose(fp);
   getch();
}
```

F. Write a C Program to copy the contents of a file to another.

```
#include<stdio.h>
#include<conio.h>
 void main()
{
 FILE *fp, *fp1;
 char ch;
 clrscr();
 fp=fopen("input", "w");
 printf("\n enter the content of the file, press cntrl+z when finish \n");
 while((ch=getchar())!=EOF)
      putc(ch, fp);
      fclose(fp);
 fp=fopen("input","r");
 fp1=fopen("output", "w");
 while((ch=getc(fp))!=EOF)
     putc(ch, fp1);
     fcloseall();
 fp1=fopen("output", "r");
 printf("the content of the copied file: ");
 while((ch=getc(fp1))!=EOF)
      printf("%c", ch);
      fclose(fp1);
 getch();
}
```