University of Sargodha

BS 1st Term Exam 2017

Subject: IT/SE **Paper: Programming Fundamentals (CMP-2122)**

Time Allowed: 2:30 Hours **Maximum Marks: 60**

Objective Part Compulsory

Q1. Attempt all parts and each require answer 2 – 3 lines (16*2=32)

1. What is the purpose of default statement in switch statement?

A switch statement can have an optional default case, which must appear at the end of the switch. The default case can be used for performing a task when none of the cases is true. No break is needed in the default case.

2. What is variable initialization?

Initializing a variable means specifying an initial value to assign to it (i.e., before it is used at all). Notice that a variable that is not initialized does not have a defined value, hence it cannot be used until it is assigned such a value.

3. Define EOF marker.

Short for end-of-file, EOF is a code placed by a computer after a file's last byte of data. EOF marks are helpful in data transmission and storage. EOF marker aids the computer to identify that it has allocated enough storage space to a file. End marker represents a state in a computer operating system when one cannot read more data from a data file or stream.

4. What is the purpose of file pointer?

File pointer is a pointer which is used to handle and keep track on the files being accessed. A new data type called "FILE" is used to declare file pointer.

5. What is built-in function? Give example.

Standard library functions are also known as built-in functions. Functions such as *puts(), gets(), printf(), scanf()* etc are standard library functions. These functions are already defined in header files (files with .h extensions are called header files such as stdio.h), so we just call them whenever there is a need to use them.

6. What is Run-Time Error? Give example.

Errors which occur during program execution(run-time) after successful compilation are called run-time errors. One of the most common run-time error is division by zero also known as Division error. These types of error are hard to find as the compiler doesn't point to the line at which the error occurs.

7. What is Dangling Pointer?

Dangling pointers in computer programming are pointers that pointing to a memory location that has been deleted (or freed).

Dangling pointers arise during object destruction, when an object that has an incoming reference is deleted or deallocated, without modifying the value of the pointer, so that the pointer still points to the memory location of the deallocated memory.

8. Differentiate between Local and Formal parameters.

Actual Parameters	Formal Parameters
When a function is called, the values (expressions) that are passed in the function call are called the arguments or actual parameters.	The parameter used in function definition statement which contain data type on its time of declaration is called formal parameter.
These are the variables or expressions referenced in the parameter list of a subprogram call.	These are the variables or expressions referenced in the parameter list of a subprogram specification.

Actual Parameters	Formal Parameters
Actual Parameters are the parameters which are in calling subprogram.	Formal Parameters are the parameters which are in called subprogram.
There is no need to specify datatype in actual parameter.	The datatype of the receiving value must be defined.
The parameters are written in function call are known as actual parameters.	The parameters are written in function definition are known as formal parameters.
Actual Parameters can be constant values or variable names.	Formal Parameters can be treated as local variables of a function in which they are used in the function header.

9. Which function is used to copy on string to another string?

The **`strcpy()`** function is a built-in library function, declared in the string.h header file. It copies the source string to the destination string until a null character (\0) is reached. Syntax:

```
strcpy(Destination, Source);
```

10. What is pointer constant?

A pointer constant is a pointer through which the value of the variable that the pointer points cannot be changed. The address of these pointers can be changed, but the value of the variable that the pointer points cannot be changed.

11. What is the drawback of Global Variables?

Too many variables declared as global, then they remain in the memory till program execution is completed. This can cause of Out of Memory issue. Data can be modified by any function. Any statement written in the program can change the value of the global variable. This may give unpredictable results in multi-tasking environments. If global variables are discontinued due to code refactoring, you will need to change all the modules where they are called.

12. What is the Infinite loop?

There may exist some loops which can iterate or occur infinitely. These are called Infinite Loop. These loops occur infinitely because their condition is always true. We can make an infinite loop by leaving its conditional expression empty (this is one of the many possible ways). When the conditional expression is empty, it is assumed to be true. To terminate an infinite loop, press Ctrl + C.

Subjective Part (4*9=36)

Note: Attempt any four questions. All questions carry equal marks.

Q2. Write down a program that reads 10 elements in an array from user. Pass the array to a method that prints the sum of given numbers.

```
#include<iostream>
int sumofarray(int a[])
 {
        int i,sum=0;
```

```
    for(i=0; i<10; i++)
    {
         sum+=a[i];
    }
        return sum;
 }
int main()
{
    int a[10],i,sum;

    printf("Enter elements in array : ");
    for(i=0; i<10; i++)
    {
        scanf("%d",&a[i]);
    }
    sum=sumofarray(a);
    printf("sum of array is :%d",sum);
}
```

Q3. Write down a program that accepts an integer number from user. Calculate and display its factorial by using a recursive function.

```
#include<stdio.h>
int factorial(int n);
int main() {
    int n;
    printf("Enter a positive integer: ");
    scanf("%d",&n);
    printf("Factorial of %d = %d", n, factorial(n));
    return 0;
}

int factorial(int n) {
    if (n>=1)
        return n * factorial(n-1);
    else
        return 1;
}
```

Q4. Write down a C program that prints the following pattern.

```
*    *    *    *    *
*    *    *    *
*    *    *
*    *
*
```

```
#include <stdio.h>
int main() {
   int i, j;
   for (i = 5; i >= 1; --i) {
      for (j = 1; j <= i; ++j) {
         printf("* ");
      }
      printf("\n");
   }
   return 0;
}
```

Q5. Write down a program that counts total number of characters in a text file.

```
#include <stdio.h>
#define MAX_FILE_NAME 100

int main()
{
    FILE* fp;
    // Character counter (result)
    int count = 0;

    char filename[MAX_FILE_NAME];

    // To store a character read from file
    char c;


    printf("Enter file name: ");
    scanf("%s", filename);

    fp = fopen(filename, "r");

    // Check if file exists
    if (fp == NULL)
    {
        printf("Could not open file %s", filename);
        return 0;
    }

    // Extract characters from file and store in character c
    for (c = getc(fp); c != EOF; c = getc(fp))
    {
        // Increment count for this character
        count = count + 1;
    }
    // Close the file
    fclose(fp);

    // Print the count of characters
    printf("The file %s has %d characters\n ", filename, count);

    return 0;
}
```

Q6. Write down a program that declares a structure Student to store rollno(int), name(string) and cgpa(float). The program defines an array to store record of five students. It inputs five students and then display the record of each student.

```
#include<iostream>

struct student
{
        int rollno;
        string name;
        float cgpa;
};

int main()
{
        struct student s1, s2, s3, s4, s5;

        printf("\nEnter details of First Student\nEnter Name:");
        scanf("%s", &s1.name);
        printf("Enter Roll No:");
        scanf("%d", &s1.rollno);
        printf("Enter CGPA:");
        scanf("%f", &s1.cgpa);

        printf("\nEnter details of Second Student\nEnter Name:");
        scanf("%s", &s2.name);
        printf("Enter Roll No:");
        scanf("%d", &s2.rollno);
        printf("Enter CGPA:");
        scanf("%f", &s2.cgpa);

        printf("\nEnter details of Third Student\nEnter Name:");
        scanf("%s", &s3.name);
        printf("Enter Roll No:");
        scanf("%d", &s3.rollno);
        printf("Enter CGPA:");
        scanf("%f", &s3.cgpa);
        printf("\nEnter details of Fourth Student\nEnter Name:");
        scanf("%s", &s4.name);
        printf("Enter Roll No:");
        scanf("%d", &s4.rollno);
        printf("Enter CGPA:");
        scanf("%f", &s4.cgpa);

        printf("\nEnter details of Fifth Student\nEnter Name:");
        scanf("%s", &s5.name);
        printf("Enter Roll No:");
        scanf("%d", &s5.rollno);
        printf("Enter CGPA:");
        scanf("%f", &s5.cgpa);


        printf("\n*****Displaying Results of First Student*****\n");
        printf("Name: %s \n", s1.name);
        printf("Roll No: %d \n", s1.rollno);
        printf("CGPA: %f \n", s1.cgpa);

        printf("\n*****Displaying Results of Second Student*****\n");
        printf("Name: %s \n", s2.name);
        printf("Roll No: %d \n", s2.rollno);
        printf("CGPA: %f \n", s2.cgpa);

        printf("\n*****Displaying Results of Third Student*****\n");
        printf("Name: %s \n", s3.name);
        printf("Roll No: %d \n", s3.rollno);
        printf("CGPA: %f \n", s3.cgpa);

        printf("\n*****Displaying Results of Fourth Student*****\n");
        printf("Name: %s", s4.name);
        printf("Roll No: %d", s4.rollno);
        printf("CGPA: %f", s4.cgpa);

        printf("\n*****Displaying Results of Fifth Student*****\n");
        printf("Name: %s \n", s5.name);
        printf("Roll No: %d \n", s5.rollno);
        printf("CGPA: %f \n", s5.cgpa);

        return 0;
}
```

Q7. Write down a program that prompts a 4-digit number from user and reverse it. For example, if user enters 2348 the program should print 8432.

```
#include <iostream>
int main()
{
    int number;
    printf("Enter a 4 digit numbe:");
    scanf("%d", &number);

    int first_digit, second_digit, third_digit, fourth_digit;

    /*Let's assume the number entered is 2348 number%10 will give 8 and thus the fourth
digit */

    fourth_digit = number%10;

    /*
    Now we have fourth digit number/10 will give 234 Again taking the remainder of the
division of 234 by 10 will give 4, the third digit
    */

    number = number/10;
    third_digit = number%10;
    number = number/10;
    second_digit = number%10;
    number = number/10;
    first_digit = number%10;

    int reversed =(fourth_digit*1000)+(third_digit*100)+(second_digit*10)+(first_digit*1);
    printf("The reversed number is %d", reversed);

    return 0;
}
```