PROGRAMMING FOR PROBLEM SOLVING

UNIT 5 ONE SHOT

WELCOME

INTELLIGENCE LEARNING

UNIT 5 BCS101 / BCS201: PROGRAMMING FOR PROBLEM SOLVING

Ques. What is pointer and how is it initialized?
Ans: Pointer is a special variable that stores the address of another variable. The pointer variable might belong to any of the data types (int, char, float).
Syntax: *(Declaration of Pointer)*
DataType *Variable_name;
Ex: int *p; → where * denotes that p is a pointer variable not a normal variable.
Initialization:
int a = 5;
int *p = &a;

Ques. What is double pointer? How will you declare double pointer in C?
Ans: When we define a pointer to pointer - the first pointer is used to store the address of the variable and the second pointer is used to store the address of first. That is why second pointer is called double pointer.
Declaration: Same as single pointer declaration.
Syntax: Data type **Variable name;
Example: int **p;

Ques. What is void pointer?

Ans: A pointer that does not have any datatype associated with it. It can be used to store address of any type of variable.

Example:-

```
int a;
float b;
char c;
void *ptr;
```

Ques. State the features of pointer. WAP to sort given number using pointer.

Ans: Features of pointer are:

- It reduces the length & complexity of program.
- It helps fast execution & better memory utilization.
- It helps to perform dynamic memory allocation & deallocation.
- It helps to build complex data structure (e.g., Linked List, Trees).

WAP to sort numbers using pointer

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

int *a, n, i, j, t;
printf("Enter how many numbers");
scanf("%d", &n);
a = (int *) malloc(n * sizeof(int));
if (a == NULL) {
    printf("Memory not available");
    exit(0);
}

for (i = 0; i < n; i++) {
    printf("Enter value");
    scanf("%d", (a + i));
}
```

```
for (i = 0; i < n - 1; i++) {
    for (j = i + 1; j < n; j++) {
        if (*(a + i) > *(a + j)) {
            t = *(a + j);
            *(a + j) = *(a + i);
            *(a + i) = t;
        }
    }
}

printf("Sorted list is:\n");
for (i = 0; i < n; i++) {
    printf("%d\t", *(a + i));
}
```

Ques. What is preprocessor directive? Explain how these are defined and called in C? Also explain types of preprocessor directive.

Ans: The difference between *#include<stdio.h>* and *#include "stdio.h"*

Preprocessor Directive

A program which preprocesses the source code before it is converted into executable code by compiler.

It uses different directives preceded by # symbol to perform its task.

Classification of Preprocessor Directives:

1. Macro substitution
2. File inclusion
3. Conditional compilation

Macro Substitution:

Macros are used to define values for symbolic constant (make use of #define preprocessor directive).

Macro template is replaced by its macro expansion before compile.

Types of macros:

Simple macro

Argument based macro

File Inclusion:
It is used to include the content of any file in any other program using #include preprocessor directive.

There are two ways to do it:
#include <stdio.h>
→ This method will search the file in standard or specific list of directories only.
#include "stdio.h“

→ This method will search the file in current directory as well as standard path.

→ Example: "stdio.h“

Conditional Compilation:
This method helps programmers to write compatible code for two different machines.

Ques. Different modes in which files can be opened with fopen() syntax?

Ans. Syntax:- FILE *fp; fp = fopen("filename", "mode");

6 Main Types of File Opening Modes:

1. "r" – Open file for reading, and file must exist.
2. "w" – Open file for writing.
   - If file doesn't exist: created.
   - If file exists: content erased.
3. "a" – Open file for appending.
   - Adds data at the end.
   - File created if not existing.
4. "r+" – Open file for reading and writing, must exist.
5. "w+" – Open file for reading and writing.
   - Created if doesn't exist.
   - Content erased if exists.
6. "a+" – Open file for appending and reading.
   - All data added at the end.
   - File created if doesn't exist.

Ques. Explain file handling library functions in C with syntax and example.

Function	Description	Example
fopen()	Used to open file	fp = fopen("a.txt", "r");
fclose()	Used to close the file	fclose(fp);
getc()	Reads single character	n = getc(fp);
fgetc()	Reads single character	n = fgetc(fp);
putc()	Writes single character	putc(ch, fp);
fputc()	Writes single character	fputc(ch, fp);
fprintf()	Prints to file using format	fprintf(fp, "%d", n);
fscanf()	Used to read from file using format	fscanf(fp, "%d", &n);

WAP in C to count the number of characters in a file and copy the text of that file to another file.

```
#include<stdio.h>
#include<conio.h>
void main() {
   FILE *fp, *f2;
   char ch;
   int c = 0;
   fp = fopen("data.txt", "r");
   f2 = fopen("file2.txt", "w");
   if(fp == NULL || f2 == NULL) {
      printf("File can't open");
      exit(1);
   }
   while((ch = fgetc(fp)) != EOF) {
      c++;
      fputc(ch, f2);
   }
   printf("The number of characters in file = %d", c);
   fclose(fp);
   fclose(f2);
}
```

A file named data.txt contains a series of integer numbers. WAP to read numbers from file and count how many are odd and how many are even, and write result to "count.txt".

```
#include<stdio.h>
#include<conio.h>
void main() {
   FILE *fp, *f2;
   int n, e = 0, o = 0;
   fp = fopen("data.txt", "r");
   f2 = fopen("count.txt", "w");
   if(fp == NULL || f2 == NULL) {
      printf("File can't open");
      exit(1);
   }
   while(fscanf(fp, "%d", &n) != EOF) {
      if(n % 2 == 0)
         e++;
      else
         o++;
   }
   fprintf(f2, "Even = %d\nOdd = %d", e, o);
   fclose(fp);
   fclose(f2);
}
```

Q: What is dynamic memory allocation, and what is its role? How does it help in building complex programs? Differentiate (1) Static & Dynamic Memory Allocation (2) malloc() & calloc().

Ans: There are 2 types of memory allocation:

1. Static Memory Allocation

Static memory allocation is done at **compile time**.

Memory allocation is **fixed** in nature.

We can't reuse the unused memory.

Easy execution & faster.

It is less efficient.

Example: Array

2. Dynamic Memory Allocation

Dynamic memory allocation is done at **run-time**.

The user can allocate memory as per the need, and also the user can release the memory when it is no longer required.

It is more efficient.

The concept of DMA enables the programmer to allocate memory at run-time.

Example: Linked List

There are 4 library functions of stdlib.h to allocate memory:

1) malloc()

2) calloc()

3) realloc()

4) free()

malloc() & calloc()→ Both functions are used to allocate memory at run time and return NULL if memory is not allocated.

malloc()

- Allocates memory in single blockInitial value is garbage.
- Use single argument.
- malloc is faster than calloc

Syntax:

ptr = (int *) malloc(n * sizeof(int));

calloc()

- Allocates memory in **multiple** blocks
- Initial value is **zero**
- Use **two arguments**
- calloc takes **little longer** than malloc

Syntax:

ptr = (int *) calloc(n, sizeof(int));

Q: What is Linked List? Write the self-reference structure of a node in Linked List. Write the advantages & disadvantages of Linked List. (2018–19)

Ans:

Linked List is also known as dynamic data structure where each structure of the list is called a **node**. The node consists of two fields one stores the **value**, and the other contains the **address of next node**.

The node is represented by:

```
struct node {
   int item;
   struct node *next;
};
```

The above structure which contains a member field that points to some structure type is called a self-referential structure.

Head → Data | Next → Data | Next → Data | Next → NULL

Advantages	Disadvantages
It can grow or shrink during execution, hence avoids memory wastage.	It consumes more time to access the arbitrary item.
It provides flexibility.	It consumes more space compared to an array.

Explain command line argument in C with example

1. It is a parameter supplied to a program when the program is invoked.
2. This parameter may represent a file name or input strings.

Syntax:

```
main(int argc, char *argv[])
{
}
```

Program to find factorial using CLA

```
#include <stdio.h>
#include <conio.h>
#include <stdlib.h>
void main(int argc, char *argv[]) {
   int n, f = 1, i;
   n = atoi(argv[1]);
   for(i = n; i > 1; i--) {
      f = f * i;
   }
   printf("Factorial = %d", f);
   getch();
}
```

THANK YOU FOR WATCHING

DO LIKE SHARE COMMENT AND SUBSCRIBE