classmate
Date 7/11/23
Page

POINTERS

Q. What is the difference between a normal variable and a pointer variable ?

Ans →

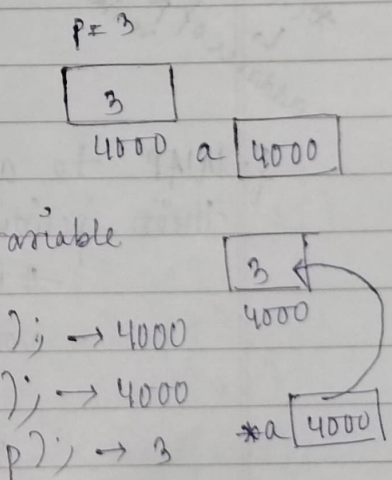
Normal Variable	Pointer Variable
	→ Pointer is a special variable that stores the address of another variable
	→ Pointer can have any name i.e. legal for any other variable.
	→ Also it is declared in same fashion like other variables but "*" is always denoted by an asterick operator (*).

Q. WAP to display the address of a variable.

```
void main()
{
    int p, *a;
    p = 3;
    a = &p;
    printf("Address of p is %u", &p);  → 4000
    printf("Address of p is %u", a);   → 4000
    printf("The value of p is %d", p);  → 3
    printf("The value of p is %d", *a); → 3
}
```

normal variable ← int p, *a; → pointer variable

↳ float and char type can't be a pointer variable



→ We can use format specifier %u or %p to print the address of a variable

→ %p format specifier in C language used to print the pointer type data. It prints the memory address of a variable in hexadecimal form. On the other hand, %u is the format specifier used to print unsigned integers. If it is used to print the memory address in unsigned integer form.

Q. WAP to print the value of the variables in different ways.

```
void main()
{
int a, *p;
a = 5;
p = &a;
printf("Address is %u", p);   → 1000
printf("value is %d, %d and %d", a, *p, *(&a));
}
```

a [5] 1000

5 5 5

*(&a) → address of pointer variable.

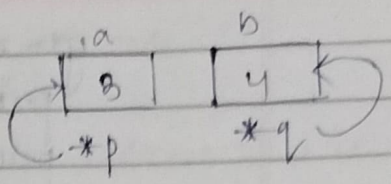
Q. WAP to add two numbers, two variables and their pointers.

```
#include<stdio.h>
void main()
{
int a, b, c;
int *p, *q;
p = &a;
q = &b;
```

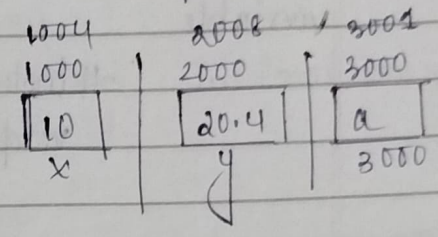
```
printf("Enter two numbers : ");
scanf("%d %d", &a, &b);
c = *p + *q;

printf("The addition is %d", c);

}
```



Q. WAP to show the effect of increment on pointer variables.




```
#include<stdio.h>
void main()
{
    int x, *x1;
    double y, *y1;
    char z, *z1;

printf("Enter the value of x, y, and z");
scanf("%d %lf %c", &x, &y, &z);
    x1 = &x;  → 1000
    y1 = &y;  → 2000
    z1 = &z;  → 3000

1000 ← printf("The address of x is %u", x1);
2000 ← printf("The address of y is %u", y1);
3000 ← printf("The address of z is %u", z1);
   4 ← printf("Size of integer is %d", sizeof(x));
   8 ← printf("Size of double is %d", sizeof(y));
   1 ← printf("Size of char is %d", sizeof(z));
    x1++;  → 1004
    y1++;  → 2008
```

```
x1++; → 3001
printf("After increment");
printf("The address of x is %u", x1); → 1004
printf("The address of y is %u", y1); → 2008
printf("The address of x is %u", z1); → 3001
}
```

Q. WAP to show the effective of increment and decrement operators used as prefix and suffix with the pointer variable.

```
#include <stdio.h>
void main()
{
int a, *pa;
printf("Enter the value of a");
scanf("%d", &a);
pa = &a;
printf("The address of a = %u", pa); → 4020
printf("The address of a = %u", ++pa); → 4024
printf("The address of a = %u", pa++); → 4024
printf("The address of a = %u", --pa); → 4024
printf("The address of a = %u", pa--); → 4024
printf("The address of a = %u", pa); → 4020
}
```

a

a

10

4020

If the memory address of a is 4020, then what will the output of the above code.

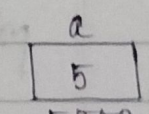
Q. What will be the output of the following C code:

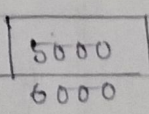
i)
```
void main()
{
int a,*pa;
a=5;
pa=&a;
printf("%u",pa);      → 5020
printf("%d",*pa);     → 5
pa++;
printf("%u",pa);      → 5024
printf("%d",*pa);     → garbage value/random value
printf("%d",a);       → 5
pa--;
printf("%u",pa);      → 5020
printf("%d",*pa);     → 5
}
```

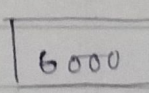
If the memory address of a is 5020, then what will be the output?

ii)
```
void main()
{
int a,*pa,*pb;
a=5;
pa=&a;
pb=&pa;
printf("%d",*pa);     → 5
printf("%d",**pb);    → 5
printf("%u",*pb);     → 5000
printf("%d",**(&pa)); → 5
}
```

If the memory address of a is 5000 and pa is 6000, then what will be the output?







Difference between Call by Value and Call by Refference.

Call by Value.

```
#include<stdio.h>
void swap(int a, intb);
void main ( )
    {

  int x,y;
  printf("Enter the value
         of x & y ");
  scanf("%d %d", &x,&y);
    swap(x,y);
  printf("The value of x & y
     are %d & %d ",x,y);
}                    ↓   ↘
                     5    10

                   5    10
void swap(int a,int b)
      {
      int c;
      c=a;
      a=b;
      b=c;
  printf("The value of
   a & b are %d & %d",
                a, b);
}            ↙   ↘
            10    5
```

Call by Refference.

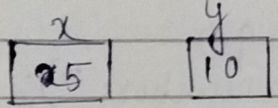
```
#include<stdio.h>
void swap(int *pa, int *pb);
void main( )
   {
 int x,y;
 printf("Enter the value of
          x & y");
 scanf("%d %d", &x,&y);
 swap(&x,&y);
 printf("The values of x & y are
 %d & %d ",x,y);
}

void swap(int*pa, int *pb)
   {
 int c;
 c=*pa;
 *pa= *pb;
 *pb = c
printf("The value of x & y
 are %d & %d ", *pa, *pb);
   }
```

x	y
5	10

Q. WAP to exchange the value of two variables.

If the value of x and y are 5 & 10, then what will be the output?



- In the call by value, the value of actual arguments is passed to the formal argument and the operation is done on the formal arguments.
- Any change in the formal argument, doesnot affect the actual argument.
- Changes made in the formal argument are local to the block of the called function.
- In call by reference, instead of passing values, the addresses are passed.
- Here, the formal arguments are pointer to the actual arguments.
- Therefore, the changes made in the formal argument are permanent.

Function returning more than one value.

Q. WAP to add and substract two numbers using the concept of call by reference.

Q. WAP to add and substract two numbers using a user defined function with two output parameter.

```
#include<stdio.h>
void add(int a, int b, int *sum, int *sub);
void main()
    {
    int x, y, p, q;
  printf("Enter two numbers");
    scanf("%d %d", x, y);
    add(x, y, &p, &q);
  printf("The addition is %d and substraction
    is %d", p, q);
}

void add(int a, int b, int *sum, int *sub)
        {
          *sum = a+b;
          *sub = a-b;
        }
```

Q1. WAP for an automatic teller machine that dispenses money. The user should enter the amount desire (a multiple of $10) and machine dispenses this amount using the least no. of bills. The bills dispensed are 50s, 20s and 10s. Write a function with 3 output parameters that determines how many each kind of bill dispense.

```
void amount(int a, int *fifty, *twenty, *ten);
```

Analysis: Here in this problem, the user will enter the amount desire and this amount will be dispensed of 50s, 20s, 10s. Here, you will use a user defined function amount() which will have 4 parameters from which 1 will be input parameter and another 3 will be output parameter.

```
#include<stdio.h>
void amount(int a, int *fifty, int *twenty, *ten);

void main()
{
int a,b,c,d;
printf("Enter the amount dispensed");
scanf("%d", &a);
amount(a, &b, &c, &d);
printf("The no. of 50s are %d, no. of 20s are %d
and no. of 10s are %d", b, c, d);
}

void amount(int a, int *fifty, *twenty, *ten)
{
```

```
*fifty = a/50;
 a = a%50

*twenty = a/20;
 a = a%20;
*ten = a/10;
 }
```

Q2. WAP to dispense change. The user enters the amount paid and the amount due. the program determines how many dollars, quarters, dimes, nickels, pennies should be given as change. Write a function with 5 output parameters that determines the quantity of each kind of coin.

Analysis: In this problem, the user will enter the amount due and amount paid and we have to determine the quantity of each kind of coin. Here we will use a user defined function with no return type and the function will have 7 parameters (2 i/p and 5 o/p).

1 dollar = 100 pennies
1 quarter = 25 pennies
1 dime = 10 pennies
1 nickel = 5 pennies

Eg: The user will enter the amount due = $1000 dollar
98 pennies
The amount paid = 200 dollar 21 pennies.
Remaining amount = 800 dollar 77 pennies.

800 dollars
3 quarters
0 dimes 0 pennies
0 nickel

21/11/23

Q. Determine the following information about each value in a list of positive integers.
a) a multiple of 7, 11 or 13.
b) is the sum of the digits odd or even?
c) a prime number

Write a function with three type int output parameters that will give the answers for above three questions.

~~#include~~

Analysis: In this problem, the user will enter an integer and it will display the three properties about the integer. (a,b,c) Here we will use user defined function check() which will have total four arguments and no returns. From those arguments, one will be the input parameter and another three will be the output parameters.

```
#include<stdio.h>
void check(int a, int *mul, int *sum, int *prime);
int main()
{
int a,b,c,d;
printf("Enter a number: ");
scanf("%d",&a);
check(a,&b,&c,&d);
return 0;
}
```

```
#include <stdio.h>
void check(int a, int *mul, int *sum, int *prime);
int main()
    {

    int a, b, c, d;
    printf("Enter a number : ");
     scanf("%d", &a);
    check(a, &b, &c, &d);
     if(b==1)
         printf("Divisible by 11, 13 or 7");
    else
         printf("Not Divisible");
     if(c==0){printf("Sum of digit is even");} else{
printf("Sum of digit is odd");
```

```
void check(int a, int *mul, int *sum, int *prime);
    {
if(a%7==0 || a%11==0 || a%13==0)
    *mul=1;
else
    *mul=0;
int s=0;
while(a!=0)
    {

int r= a%10;
*sum= *sum+r; s=s+r;
a=a/10;
if (*sum % 2==0)
    *sum=0;
else
*sum=1; }
```

```
if(d==0)
  printf("Prime");
else
  printf("Not prime");

return 0;
}
```

```
for(int i=2; i<=a/2; i++)
    {
        int c=0;
    if(a%i==0)
      {
      *prime=0; c++;
      break;
      }
    else
        if(c==2)
      *prime=0;      *prime=1;
                  else
                      *prime=0;
    }
```

25/11/23

Q1 WAP to take two numerical list of the same length and store the list in arrays x and y, each of which has 20 elements. Let n be the actual number of the data values in each list. Store the product of the corresponding elements of x and y in the third array z, also of size of 20. Display the arrays x, y and z in 3 column table. Then compute and display the square root of the sum of the items in z.

Q2. WAP to take two numerical list of the same length ended by a sentinel value and store the list in arrays x & y, each of which has 20 elements. Let n be the actual no. of data values in each list. ...
So rest as Q1.

1.

```
#include<math.h>
#include<stdio.h>
int main()
{
    int x[20]; //int x[] = {1,2,3...}
        y[20],z[20];
printf("Enter the no. of elements in x&y");
 scanf("%d",&n); printf("Enter the elements in array x");
 for(int i=0;i<n;i++)
        scanf("%d",&x[i]);
 printf("Enter the elements in array y");
     for(i=0;i<n;i++)
            scanf("%d",&y[i]);

 for(i=0;i<n;i++)
    {
     z[i] = x[i]+y[i];
    }

 printf("Array x : array y : array z");
 for(i=0;i<n;i++)
        printf("%d%d%d",x[i],y[i],z[i]);

 int sum=0;
 for(i=0;i<n;i++)
        sum = sum+z[i];
 printf("The square of the sum of the
        items in array z is %lf", sqrt(sum));

    return 0;
    }
```

2.

```
#include<stdio.h>
#include<math.h>
#define sentinel -1.
int main()
{

int x[20], y[20], z[20], i=0, n;
printf("Enter the 1st element of array x");
scanf("%d", &x[i]); printf("Enter the elements of array x -1 to stop)");
while(x[i]!=sentinel && i<20)
{
scanf("%d", x[++i]);
}
n=i;
printf("Enter the elements of array y");
for(int a=0; a<n; a++)
scanf("%d", y[a]);
```

Q3. Write a C program can represent a real polynomial $p(x)$ of degree n as an array of the real coefficients $a_0, a_1, a_2, \ldots a_n$. $p(x) = a_0 + a_1 x + a_2 x^2 + \ldots a_n x^n$. Write that inputs a polynomial of max. degree & ~~that~~ and then evaluates the polynomial at various values of x. Include a function get_poly that fills the array of coefficients and sets the degree of the polynomial and sets the degree of x. Include another function eval_poly that evaluates a polynomial at a given value of x.

prototype

```
void get_poly(double coeff[], int *degree);
double eval_poly(double coeff[], int degree,
                 double x);
```

```
#include<stdio.h>
#include<math.h>
void get_poly(double coeff[], int * degree);
double eval_poly(double coeff[], int degree, double x);
void main()
{
double coeff[9];
int degree;
double x;
get_poly(coeff, &degree);
printf("Enter the value of x");
scanf("%lf", &x);
double poly = eval_poly(coeff, degree, x);
printf("The polynomial value is %f",
                poly);
}
```

```
void get_poly(double coeff[], int * degree)
{
    printf("Enter the degree");
    scanf("%d", degree);
    printf("Enter the real coefficients");
    for(int i=0; i<=*degree; i++)
        scanf("%lf", &coeff[i]);
}

double eval_poly(double coeff[], int degree,
                 double x)
{
    double poly, sum=0;
    for(int i=0; i<=degree; i++)
    {
        poly = coeff[i] * pow(x,i);
        sum = sum + poly;
    }
    return sum;
}
```

Q4. A normalized vector X is defined as $X_i = \frac{V_i}{\sqrt{\sum_{i=1}^{n} V_i^2}}$ where $i = 1, 2, 3 \ldots n$. Design a tested program that normalizes vector of different length. Define three functions scan_vector, normalize_vector and print_vector

```
#include <stdio.h>
#include <math.h>
int scan_vector(double vector[]);
void normalized_vector(double vector[], double x[],
                       int size);
void print_vector(double x[], int size);
void main()
{
    double vector[100];
    int size = scan_vector(vector);
    double x[100];
    normalized_vector(vector, x, size);
    print_vector(x, size);
}

int scan_vector(double vector[])
{
    int n;
    printf("Enter the size (n)");
```

```
    scanf("%d", &n);
for(int i=0; i<n; i++)
    {
    scanf("%lf", &vector[i]);
}

return n;
}

void normalize_vector(double vector[], double x[],
                      int size)
    {  double
       int root, sum;
for(int i=0; i<size; i++)
    {

  double sum = sum + (vector[i] * vector[i]);
}

 root = sqrt(sum);
 for(int i=0; i<size; i++)
    {
     x[i] = vector[i]/root;
    }

void print_vector(double x[], int size)
    { printf("The output vector is \n");
for(int i=0; i<size; i++)
    printf("%lf", x[i]);
}
```

Q5. WAP to search an element in an array using binary search.

Q6. WAP to sort the elements of an array using bubble sort

~~Mid Sem~~

~~Q1. WAP that would~~
~~a) prompt~~

Ch - 1, 2, 3, 4.
Ass 1, Ass 2, Ass 3, Ass 4.

Date 19/12/23
Page

STRINGS

→ In language C, a string is defined as an array of characters.

→ Every string is terminated with '\0' a null character.

→ The compiler automatically puts null at the end of character array or string.

Declaration And Initialization Of String

Q. WAP to enter a name and display it.

```
#include<stdio.h>
void main()
{
    char name[100];
    printf("Enter your name");
    scanf("%s", name);
    printf("The entered name is %s", name);
}
```

OR

```
#include<stdio.h>
void main()
{
    char name[100];
    printf("Enter your name");
    gets(name);
```

```
printf("The entered name is %s", name);
}
```

Q. WAP to initialize an character array/string and use the while loop to display the elements of string with the help of null character.

```
#include<stdio.h>
void main()
{
char name[] = "My name";
int i=0;
while(name[i] != '\0')
{
printf("%c", name[i]);
i++;
}
}
```

String Predefined functions (#include<string.h>)

predefine

① strlen(): This function determines the length of the string. ('\0' is not counted)

Q. WAP to count the no. of characters in a given string.

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

```
{
char name[100];
int length;
printf("Enter your name");
gets(name);
length = strlen(name);
printf("The length of the string is %d",
                                length);
}
```

(2) isspace( ) : This predefine function will check if a character is a white space character or not. (ctype.h).

→ If a character is a white space character then this predefine function will return a non zero value.

→ If a character is not a white space character then this function will return a zero.

Q. Write and test a function deblank( ) that takes a string input and string output argument and returns the copy of the input argument with all blank space removed.

```
#include <stdio.h>
#include <ctype.h>
void deblank(char input[], char output[])
void main( )
{
char input[100];
```

```
char output[100];
printf("Enter a string : ");
    gets(input);
deblank(input, output);
printf("The string without blank space is %s",
                        output);

}
void deblank(char input[], char output[])
    {
    int i, j;
    char ch;
    i = 0, j = 0;
    while((ch = input[i]) != '\0')
        {
        if(!isspace(ch))
            {
            output[j] = ch;
            j++;
            }
        i++;
        }
    }
```

③ strcpy( ): #include<string.h>.
This predefine function copies a string from a source to destination.

(4) strncpy(): This predefine function copies character of a string to another string upto the specified length.

Q. WAP to copy the contents of one string to another string using strcpy().

```
#include<stdio.h>
#include<string.h>
void main()
{
    char input[100];
    char output[100];
    printf("Enter the string: ");
    gets(input);
    strcpy(output, input);
    printf("the entered string is %s", output);
}
```

Q. Write and test a function named hydroxide that returns a 1 for true if its string argument ends with the substring OH.

```
#include <string.h>
#include <stdio.h>
int hydroxide(char compound[]);
void main()
{
char compound[100];
printf("Enter chemical compound");
gets(compound);
int a = hydroxide(compound);
if(a)
  printf("%s is a hydroxide compound",
                compound);
else
  printf("%s is not a hydroxide compound",
                compound);
}
int hydroxide(char compound[])
{
  int len;
  char oh[3];
  len = strlen(compound);
  strncpy(oh, &compound[len-2], 2);
  int a = strcmp(oh, "OH");
  return(!a);
}
```

Recursion : When a function calls itself.

Q. WAP that takes a function without blanks & punctuation characters. Also in the program write a recursive function that returns a value 1 if the string argument is palindrome

```
#include<stdio.h>
#include<string.h>
int palindrome(char str[], int i, int j);
    void main()
        {

    char str[100];
  printf("Enter a string without blanks and
                        punctuation symbol");
    gets(str);
 if(palindrome(str, 0, strlen(str)-1))
        printf("%s is a palindrome", str);
 else
    printf("%s is not a palindrome", str);
}
int palindrome(char str[], int i, int j)
        {
    if(str[i] != str[j])
        return 0;
    if(i > j)
        return 1;
  return(palindrome(str, i+1, j-1));
}
```

Q. Write and test a recursive function that returns the value of the following recursive definitions.

$f(x) = 0$ if $x \leq 0$

and $f(x) = f(x-1) + 2$ otherwise.

What set of numbers is generated by this definition

Q. WAP to compare two strings upto a specified length.

Q. WAP to append the 2nd string at the end of the 1st string.

Q. WAP to append the 2nd string with specified (n) no. of characters at the end of the 1st string.

Q. WAP to display the reverse of an input string.

2)

```
#include<stdio.h>
#include<string.h>
void main()
{
char str[100];
char str1[100];
printf("Enter the 1st string");
gets(str);
printf("Enter the 2nd string");
gets(str1);
int n;
printf("Enter length upto which
        comparison will happen");
scanf("%d",&n);
int a;
a = strncmp(str, str1, n);
```

```
if(a==0)
    printf("The two strings are identical upto
                                %d characters",n);
else
    printf("The two strings are not identi-
            cal upto %d characters",n);
}
```

OUTPUT

```
Enter the 1st string
HELLO
Enter the 2nd string
HE IS
```

The two strings are identical upto 2 characters.

3)

```
#include<stdio.h>
#include<string.h>
void main()
{
char str[100];
char str1[100];
printf("Enter the 1st string");
  gets(str);
printf("Enter the 2nd string");
  gets(str1);
  strcat(str,str1);
printf("The necessary string is %s",str);
}
```

OUTPUT:

```
Enter 1st string
HELLO
Enter 2nd string
RAM
```

Necessary string is HELLORAM.