**Programming Questions:**

**1. Variables, Constants, Data Types, Operators**

**Question:**   
Write a program to calculate the compound interest using the formula:   
A=P×(1+r/n)n⋅tA=P×(1+r/n)n⋅t

Where:

* **P** is the principal amount (input by the user).
* **r** is the annual interest rate (input as a percentage).
* **n** is the number of times interest is compounded per year (constant).
* **T**   is the time in years (input by the user).

Use appropriate data types for variables and constants. Ensure the program handles invalid inputs gracefully.

#include<stdio.h>

#include<math.h>

int main()

{

double P,r,T,A;

const int n=4;

//Input principal

printf("Enter the principal amount(p): ");

if(scanf("%lf",&P)!=1 || P<=0)

{

printf("Invalid input! amount must be positive.\n");

return 1;

}

//Input rate of interest

printf("Enter the rate of interest in percentage: ");

if(scanf("%lf",&r)!=1 || r<=0)

{

printf("Invalid input! it must be positive");

return 1;

}

//Input time in years

printf("Enter the time in years: ");

if(scanf("%lf",&T)!=1 || T<=0)

{

printf("Invalid input! time must be positive");

return 1;

}

//Converting rate from percentage to decimal

r=r/100.0;

//Compound Interest Formula : A=P\*(1+r/n)^(n\*T)

A= P \* pow((1+r/n),n\*T);

printf("\n---------Compound Interest Calculation-------------\n");

printf("Principal : %.2lf\n",P);

printf("Rate of interest : %.2lf%%\n",r\*100);

printf("Compounded : %d times per year\n",n);

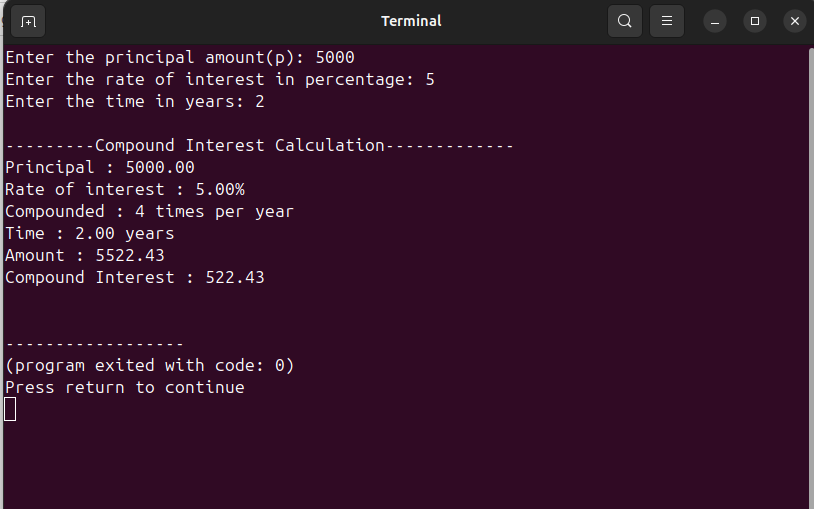
printf("Time : %.2lf years\n",T);

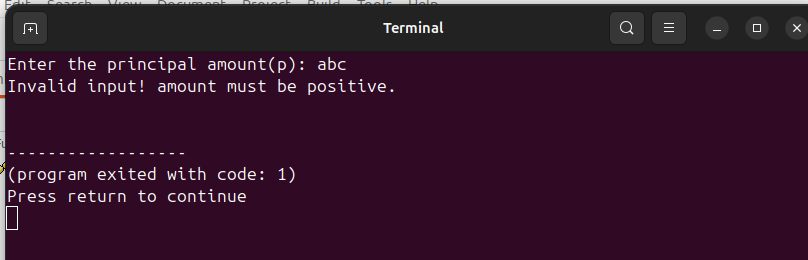
printf("Amount : %.2lf\n",A);

printf("Compound Interest : %.2lf\n",A-P);

return 0;

}





**2. User-Defined Data Types and Storage Classes**

**Question:**   
Create a program that defines a typedef for a structure representing a **student** with the following fields:

Name (string)

* Roll number (integer)
* Marks in 3 subjects (array of floats)

Write a function to calculate the average marks of the student and display the details. Use appropriate storage classes for variables.

CODE:

#include<stdio.h>

typedef struct {

char name[50];

int rollno;

float marks[3];

} Student;

float calculateAverage(Student s)

{

//auto is default storage class

auto float sum=0.0;

for(int i=0;i<3;i++)

{

sum+=s.marks[i];

}

return sum/3;

}

void displayStudent(Student s)

{

static int count=0; //static variable to count

count++;

float avg=calculateAverage(s);

printf("\n-----------Student details (%d)----------\n",count);

printf("Name : %s\n",s.name);

printf("Roll No : %d\n",s.rollno);

printf("Marks : %.2f,%.2f,%.2f\n",s.marks[0],s.marks[1],s.marks[2]);

printf("Average : %.2f\n",avg);

}

int main()

{

Student s1;

//Input Details

printf("Enter Student Name: ");

scanf("%s",s1.name);

printf("Enter the rollno: ");

scanf("%d",&s1.rollno);

printf("Enter the marks of 3 subjects: ");

for(int i=0;i<3;i++)

{

printf("Subject %d: ",i+1);

scanf("%f",&s1.marks[i]);

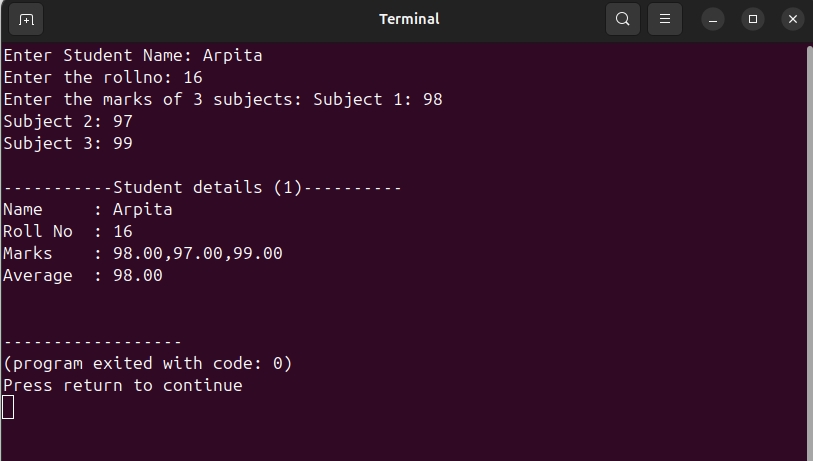
}

//Dispaly results

displayStudent(s1);

return 0;

}

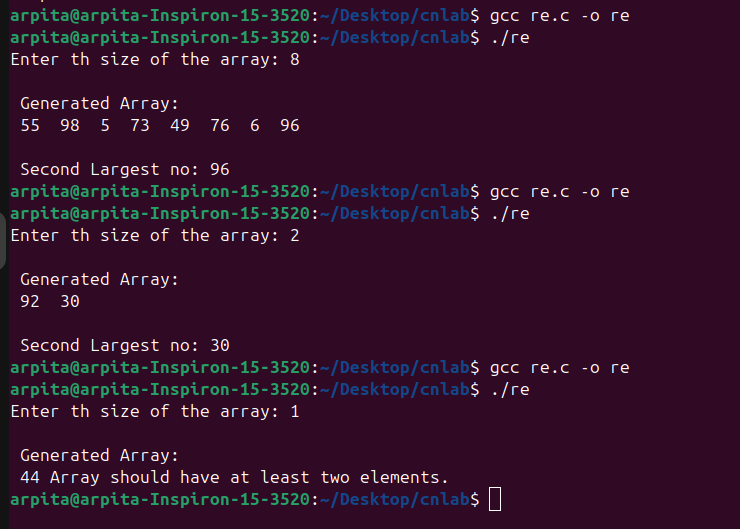


**3. Pointers, Control Statements, Functions**

**Question:**   
Write a program to dynamically allocate memory for an array of integers using pointers. The program should:

1. Accept the size of the array from the user.

1. Populate the array with random numbers between 1 and 100.
2. Use a function to find the second largest number in the array.
3. Free the allocated memory after use.



**4. Structures and Unions**

**Question:**   
Create a program that uses a **union** to store either an integer, a float, or a character. The program should:

1. Accept the type of data the user wants to store (integer, float, or character).

2. Store the value in the union and display it.

3. Explain the memory usage of the union in the program output.

#include<stdio.h>

#include<string.h>

union Data

{

int i;

float f;

char c;

};

int main()

{

union Data data;

char type[10];

printf("Enter the type of data to store(int,float,char): ");

scanf("%s",type);

if(strcmp(type,"int")==0)

{

printf("Enter an integer value: ");

scanf("%d",&data.i);

printf("you entered integer: %d\n",data.i);

}

else if(strcmp(type,"float")==0)

{

printf("Enter a float value: ");

scanf("%f",&data.f);

printf("you entered float: %.2f\n",data.f);

}

else if(strcmp(type,"char")==0)

{

printf("Enter a character: ");

scanf(" %c",&data.c);

printf("you entered character: %c\n",data.c);

}

else

{

printf("Invalid type entered!\n");

return 1;

}

//Displaying memory information

printf("\n------Memory Usage-------\n");

printf("Size of union: %lu bytes\n", sizeof(data));

printf("Size of int: %lu bytes\n", sizeof(data.i));

printf("Size of float: %lu bytes\n", sizeof(data.f));

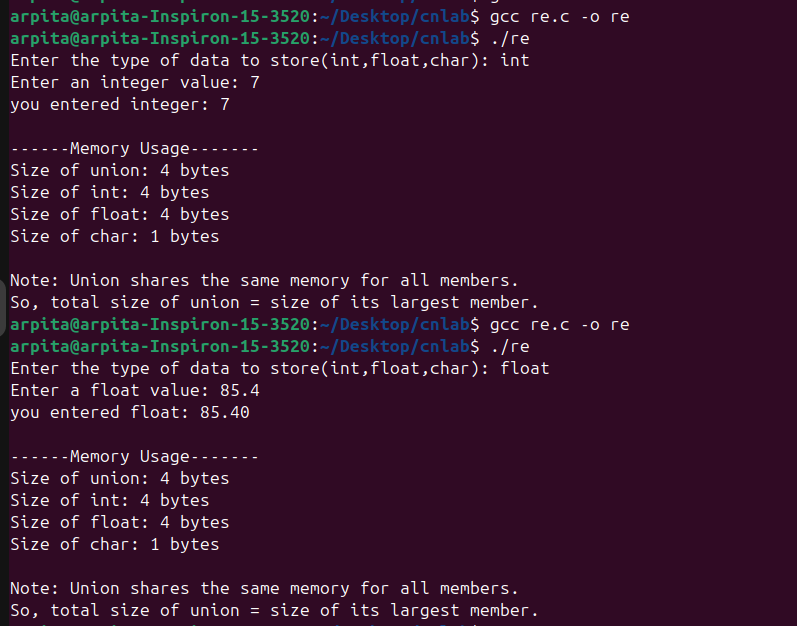
printf("Size of char: %lu bytes\n", sizeof(data.c));

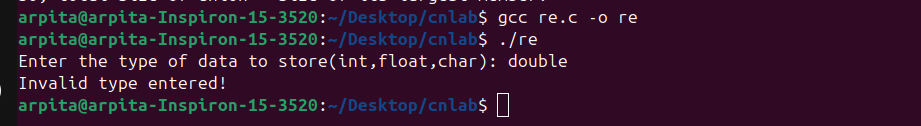
printf("\nUnion shares the same memory for all members.\n");

printf("So, total size of union = size of its largest member.\n");

return 0;

}





**5. Arrays and Strings**

**Question:**   
Write a program to check if two strings are **anagrams** of each other.

\*.An anagram is a word or phrase formed by rearranging the letters of another, such as "listen" and "silent".

\*.Ignore case and spaces while checking.

\*.Use functions for modularity (e.g., one function to sort the strings, another to compare them).

CODE:

#include<stdio.h>

#include<string.h>

#include<ctype.h>

void cleanstring(char \*str)

{

int i,j=0;

for(i=0;str[i]!='\0';i++)

{

if(str[i]!=' ')

{

str[j++]=tolower(str[i]);

}

}

str[j]='\0';

}

void sortstring(char \*str)

{

int i,j;

char temp;

int len=strlen(str);

for(i=0;i<len-1;i++)

{

for(j=i+1;j<len;j++)

{

if(str[i]>str[j])

{

temp=str[i];

str[i]=str[j];

str[j]=temp;

}

}

}

}

int anagrams(char \*str1,char \*str2)

{

cleanstring(str1);

cleanstring(str2);

if(strlen(str1)!=strlen(str2)) //if length differ

return 0;

sortstring(str1);

sortstring(str2);

if(strcmp(str1,str2)==0)

return 1;

else

return 0;

}

int main()

{

char str1[100],str2[100];

printf("Enter the first string: ");

fgets(str1,sizeof(str1),stdin);

str1[strcspn(str1,"\n")]='\0';

printf("Enter the second string: ");

fgets(str2,sizeof(str2),stdin);

str2[strcspn(str2,"\n")]='\0';

if(anagrams(str1,str2))

{

printf("The strings are anagrams.\n");

}

else

{

printf("The strings are not anagrams.\n");

}

return 0;

}

