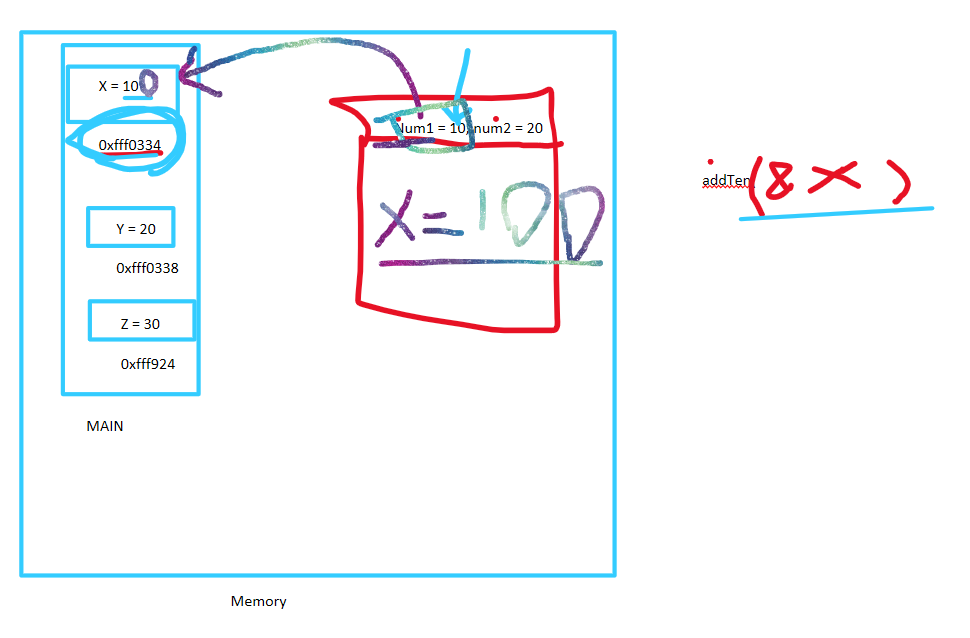
LIVESTREAM 11-02-2024

Channel: NeuroCode

Return Types in Functions

1. Void
   1. #include <iostream>
   2. using namespace std;
   3. //void
   4. // menu based
   5. void printNum(int x)
   6. {
   7. cout<<"Number is: "<<x<<endl;
   8. cout<<"printing the number again, it's value is : "<<x<<endl;
   10. return;
   11. cout<<"printing the number again, it's value is : "<<x<<endl;
   12. cout<<"printing the number again, it's value is : "<<x<<endl;
   13. }
   14. int main()
   15. {
   16. int x = 10, y = 20;
   17. printNum(x);
   18. printNum(y);
   19. return 0;
   20. }
2. Int
   1. #include <iostream>
   2. using namespace std;
   3. int calcSum(int num1,int num2)
   4. {
   5. int ans = num1+num2;
   6. return ans;
   7. }
   8. int main()
   9. {
   10. int x=5, y=10;
   11. cout<<"Sum of 5 and 10 is: "<<calcSum(x,y);
   12. return 0;
   13. }
3. Float
   1. #include <iostream>
   2. using namespace std;
   3. float divNum(float num1, float num2)
   4. {
   5. return (num1/num2);
   6. }
   7. int main()
   8. {
   9. float x=5.5,y=2.2;
   10. cout<<"division is: "<<divNum(x,y)<<endl;
   11. return 0;
   12. }
4. Bool
   1. #include <iostream>
   2. using namespace std;
   3. bool isEligible(int age)
   4. {
   5. if(age>15)
   6. {
   7. return 1;
   8. }
   9. else
   10. {
   11. return 0;
   12. }
   13. }
   14. int main()
   15. {
   16. //program to check if user is allowed to create
   17. //twitter account
   18. //condition: age should be greater than 15
   19. int age= 20;

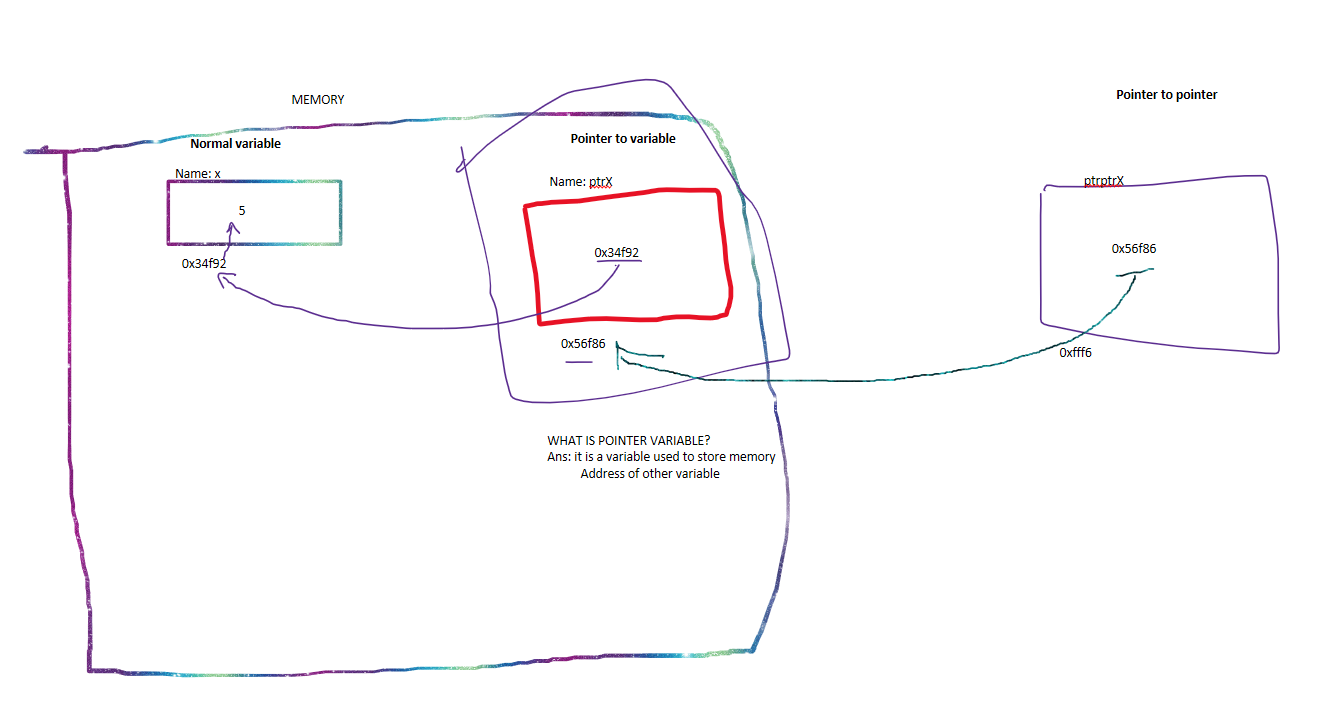



   25. if( isEligible(age) )
   26. {
   27. cout<<"You are allowed to create twitter account!"<<endl;
   28. }
   29. else
   30. {
   31. cout<<"You are not allowed to create twitter account"<<endl;
   32. }
   33. return 0;
   34. }
5. Char
   1. #include <iostream>
   2. using namespace std;
   3. //take two characters as input and print all characters
   4. //between these two characters
   5. void printCharRange(char start, char endd)
   6. {
   7. for(char i = start; i<= endd ; i++)
   8. {
   9. cout<<i<<" ";
   10. }
   11. }
   12. int main()
   13. {
   14. char x,y;
   15. cout<<"Enter Starting Range Character: ";
   16. cin>>x;
   17. cout<<"Enter Ending Range Character: ";
   18. cin>>y;
   19. printCharRange(x,y);
   20. return 0;
   21. }
6. PASS BY VALUE:
   1. Pass by value in C++ means that when a function is called, the arguments passed to that function are copied into new variables for use within the function. These new variables are local to the function, meaning that any changes made to them within the function do not affect the original arguments used to call the function.\
   2. #include <iostream>
   3. using namespace std;
   4. int calcSum(int x, int y)
   5. {
   6. cout<<"address of x in function: "<<&x<<endl;
   7. cout<<"address of y in function: "<<&y<<endl;
   8. x = 50;
   9. cout<<"value of x in function is : "<<x<<endl;
   10. return x+y;
   11. }
   12. int main()
   13. {
   14. int x=10,y=20,z=30;
   15. cout<<"address of x in main: "<<&x<<endl;
   16. cout<<"address of y in main: "<<&y<<endl;
   17. calcSum(x,y);
   18. cout<<"value of x in main is : "<<x<<endl;
   19. return 0;
   20. }
7. Pass by reference
   1. Pass by reference in C++ means that when a function is called, instead of passing copies of the arguments to the function, you pass the actual references (or addresses) of those arguments. This means the function can directly access and modify the original variables used to call the function.
   2. #include <iostream>
   3. using namespace std;
   4. void addTen(int &num1)
   5. {
   6. num1 = 150;
   7. }
   8. int main()
   9. {
   10. int x = 10;
   11. addTen( x );
   12. cout<<"value of x after using addTen( &x ) is: "<<x<<endl;
   13. return 0;
   14. }
   15. 
8. Arrays as pass by referencwe
   1. #include <iostream>
   2. using namespace std;
   3. //1. jab bhi hum array pass krty hain functions my,
   4. // to sath my hum uska size bhi pass krty hain
   5. //2. arrays are always passed by reference in functions
   6. void incrementArrTen(int arr[],int n)
   7. {
   8. for(int i=0;i<n; i++)
   9. {
   10. arr[i] = arr[i] + 10;
   11. }
   12. }
   13. int main()
   14. {
   15. int arr[5]= {10,20,30,40,50};
   16. incrementArrTen(arr,5);
   17. cout<<"values of array after calling increment function are: "<<endl;
   18. for(int i=0; i<5; i++)
   19. {
   20. cout<<arr[i]<<" ";
   21. }
   22. return 0;
   23. }

PRACTICE PROBLEMS:

1. Function to find prime number
   1. #include <iostream>
   2. using namespace std;
   3. void checkPrime(int num)
   4. {
   5. int numOfDiv = 0;
   6. for(int i=1; i<=num; i++)
   7. {
   8. if(num % i == 0)
   9. {
   10. numOfDiv++;
   11. }
   12. }
   13. if(numOfDiv == 2)
   14. {
   15. cout<<num<<" is a prime number"<<endl;
   16. }
   17. else
   18. {
   19. cout<<num<<" is not a prime number"<<endl;
   20. }
   21. }
   22. int main()
   23. {
   24. checkPrime(7);
   25. return 0;
   26. }
2. PRIME BETWEEN RANGE
   1. #include <iostream>
   2. using namespace std;
   3. //find prime numbers between a given range
   4. void checkPrimeRange(int start, int endd)//st = 10, end = 15
   5. {
   6. int numOfDiv;
   7. for(start; start <= endd ; start++)
   8. {
   9. numOfDiv =0;
   10. for(int j = 1; j<= start ; j++)
   11. {
   12. if(start%j == 0)
   13. {
   14. numOfDiv++;
   15. }
   16. }
   17. if(numOfDiv==2)
   18. {
   19. cout<<start<<" is a prime number"<<endl;
   20. }
   21. }
   22. }
   23. /\*
   24. METHOD TWO
   25. void checkPrimeRange(int start, int endd)//st = 10, end = 15
   26. {
   27. int numOfDiv;
   28. for(int i = start; i<= endd ; i++)
   29. {
   30. numOfDiv =0;
   31. for(int j = 1; j<= i ; j++)
   32. {
   33. if(i%j == 0)
   34. {
   35. numOfDiv++;
   36. }
   37. }
   38. if(numOfDiv==2)
   39. {
   40. cout<<i<<" is a prime number"<<endl;
   41. }
   42. }
   43. }
   44. \*/
   45. void checkPrime(int num)
   46. {
   47. int numOfDiv = 0;
   48. for(int i=1; i<=num; i++)
   49. {
   50. if(num % i == 0)
   51. {
   52. numOfDiv++;
   53. }
   54. }
   55. if(numOfDiv == 2)
   56. {
   57. cout<<num<<" is a prime number"<<endl;
   58. }
   59. else
   60. {
   61. cout<<num<<" is not a prime number"<<endl;
   62. }
   63. }
   64. int main()
   65. {
   66. checkPrimeRange(5,20);
   67. return 0;
   68. }
3. Factorial Function
   1. #include <iostream>
   2. using namespace std;
   3. void factorial(int num)
   4. {
   5. int fact = 1;
   6. for(int i=1;i<=num; i++)
   7. {
   8. fact= fact\* i;
   9. }
   10. cout<<"Factorial of Number is: "<<fact<<endl;
   11. }
   12. /\*
   13. //user entered 3;
   14. //for looP:
   15. i=1 fact = 1\*1; fact = 1;
   16. i=2 fact = 1\*2; fact = 2;
   17. i=3 fact = 2\*3; fact = 6;
   19. \*/
   20. int main()
   21. {
   22. factorial(10);
   23. return 0;
   24. }

**POINTERS**

****

1. POINTER
   1. #include <iostream>
   2. using namespace std;
   3. int main()
   4. {
   5. int x = 20;
   6. int \*ptrX = &x;
   7. cout<<" address of x is: "<< &x<<endl;
   8. cout<<"value in ptrX is : "<< ptrX<<endl;
   9. cout<<"address of ptrX is: "<<&ptrX<<endl;
   10. cout<<"deferencing ptrX: "<< \*ptrX<<endl;
   11. //how to change value of x using pointer?
   12. \*ptrX = 100;
   13. cout<<"value of x after updating through pointer: "<<x<<endl;
   14. return 0;
   15. }
2. POINTER TO POINTER
   1. #include <iostream>
   2. using namespace std;
   3. int main()
   4. {
   5. int x = 20;
   6. int \*ptrX = &x;
   7. int \*\*ptrptrX = &ptrX;
   8. cout<<" address of x is: "<< &x<<endl;
   9. cout<<"value in ptrX is : "<< ptrX<<endl;
   10. cout<<"address of ptrX is: "<<&ptrX<<endl;
   11. cout<<"value of ptrptrX is : "<< ptrptrX<<endl;
   12. cout<< \*(\*ptrptrX)<<endl;
   13. return 0;
   14. }
3. **Swap function using pointers as pass by reference**
   1. #include <iostream>
   2. using namespace std;
   3. //SECOND METHOD OF PASS BY REFERENCE
   4. // swap value of variables using pointers
   5. void swapNum(int \*x, int \*y)
   6. {
   7. int temp;
   8. temp = \*x;
   9. \*x = \*y;
   10. \*y = temp;
   11. }
   12. int main()
   13. {
   14. int num1=10, num2 = 30;
   15. cout<<"before swapping: num1: "<<num1<<" num2: "<<num2<<endl;
   16. swapNum(&num1, &num2);
   17. cout<<"after swapping: num1: "<<num1<<" num2: "<<num2<<endl;
   18. return 0;
   19. }