Vidhi verma

end

Display sum

Sum=n1+n2

Read n1 and n2

Declare variables n1 , n2 and sum

Start

Roll no.

Homework

Vidhi verma

BCA-B

Roll no. 41221184

Q1. Write an algorithm to find factorial of a number.

Solution:- step1:- Start

Step2:- Declare variable num1, fact i

Step3:- Read num1

Step4:- Fact=1 and i=1

Step5:- Repeat until i<=num1

Fact=Fact\*i

i=i+1

Step6:- Display Fact

Step7:- Stop.

Start

Declare variable num1, fact i

Read num1

i<=num1

NO

YES

Fact=Fact\*i i=i+1

Display fact

stop

Q2. Write an algorithm to display an integer received by user.

Solution:- step1:-start

Step2:- Declare variable num1 , Result

Step3:- Read num1

Step4:- store the value of num1 in Result

Step5:- Display Result

Step6:- stop.

start

Declare variable num1 , Result

Read num1

store the value of num1 in Result

Display Result

stop.

Q3. Write an algorithm to find out the cubic of a received integer.

Solution: - step1: - start

Step2: - Declare variable num1, Result

Step3: - read num1

Step4: -multiply

num1\*num1\*num1

Step5: -store the value of step4 in Result.

Step6: -display Result.

Step7: -stop.

start

Declare variable num1, Result

read num1

multiply num1\*num1\*num1

display Result.

stop.

Q4.Draw a flow chart of Algorithm for square of number.

Solution:-

start

Declare variable num1, Result

read num1

multiply num1\*num1

display Result.

stop.

Q5. what is the difference between a good program and a good programming?

Solution: - First of all let us understand what is good program and good programming.

So, a good program is basically a program that is time and memory efficient and which is reliable, accurate, flexible, and easy to understand.

and a good programming is basically the process of developing or building a good program.

DIFFERENCE: -



|  |  |
| --- | --- |
| Good Program | Good programming |
| 1. Short and understandable 2. It should be accurate and reliable. 3. A good program should be in a proper syntax. 4. It is the result of a good programming. | 1. The process of creating a simple and understandable program in a time efficient manner. 2. A good programming eliminates errors by handling exceptions for accurate and reliable results. 3. A good programming should follow the syntax with comment lines for better understanding. 4. It is the process of creating a good program. |

Date-23/dec/2021

**MAT**rix

**LAB**oratory

MATRIX LABORATORY

Mathematics A place that is used

for research, testing

and experiments.

* MATLAB is multi-paradigm (procedural) programming language.
* MATLAB allows matrix (mathematics) manipulations like plotting of functions, implementations of algorithms etc.
* MATLAB is basically designed as it turns everything in familiar mathematic notations.

06/Jan/2022 Homework

**Q2. Write an Algorithm, pseudocode, and flowchart to enter a number and find the reverse of the number entered by the user.**

**For understanding the logic let us take an example.**

For example, user entered a value 12

Dividing it 10

To find the remainder;

12%10= 1 and remainder will be 2

Reverse= 0\*10+2

Reverse=2

Reverse=2\*10+1 (10%1=0 and remainder will be 1)

Reverse=21

**Ans: - Algorithm**

Step1: - start

Step2: - declare variable num, rev=0, remainder

Step3: - read value of num1

Step4: - apply while loop, divide num with 10 and multiply the remainder with rev\*10

step5: - display rev

step6: - stop

**Pseudocode**

Int main()

Printf(“Enter an integer:”)

Reverse = 0

While(num!=0) then

remainder = num%10

rev=(rev\*10)+remainder

num=num/10

end while

output “Reversed number is”

**Flowchart**

start

NO

YES

read value of num1

num= num/10

stop

remainder = num%10

rev=(rev\*10) +remainder

While

(num!=0)

declare variable num, rev=0, remainder

Display rev

**Q1: - write an Algorithm, pseudocode, and flow chart to print the prime number from 1 to 100.**

To get the prime number from 1 to 100, first we have to declare both values.

First for loop Condition will be (num1<=100)

Second for loop condition will be (j<=i) where both j=1 and i++ are iterations.

If i%j = 0 then the count will increase and the process will go on.

If count==0 then the integer is prime and it will be displayed.

**Ans: - Algorithm**

Step1: -start

Step2: -declare variable num1, i, j

Step3: - num1=2

Step4: - start for loop (num1<=100) to iterate the number from 1 to num1.

Step5: -if number is prime then print it, else ignore that number and iterate next

number.

Step6: -stop.

**Pseudocode**

Int num1, i, j

Printf(“prime numbers between 1 to 100)

num1=2

for(num1=2;num1<=100;++num1)

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

if(i%j==0) then count++

if(count==2) then

printf(“%d”,i)

else, ignore the integer.

**Flowchart**

NO

start

declare variable num1, i, j

num1=2

num1=2;num1<=100;++num1

NO

num1=num1+2

YES

num1=2;num1<=100;++num1

NO

YES

i%j==0

Count++

NO

Count==2

stop

Display i

**Q3. Write an Algorithm, pseudocode, and flowchart to enter a number and test whether it is a Fibonacci number.**

Fibonacci number is the number which is the sum of its previous two numbers.

**Ans= Algorithm.**

Step1: -start

Step2: -declare variable i=2, num1, a=0, b=1, sum=0

Step3: - read value of num1

Step4: -the Fibonacci series is a+b

Step5: -if (i<=num1) then sum of a b , where a=b and b=sum.

Step6: -increment of i will occur

Step7: -print sum

Step8: -stop

**Pseudocode.**

Int main()

Printf(“enter a value”)

a=0, b=1, sum=0

(i<=num1)

Sum=a+b

a=b

B=sum, i++

printf(“%d”,sum)

**Flowchart.**

start

declare variable i=2, num1, a=0, b=1, sum=0

Read num1

the Fibonacci series is a+b

i=2

NO

YES

i%j==0

sum=a+b

a=b

b=sum, i++

Print sum

start

**Q3. Write an Algorithm, pseudocode, and flowchart to accept any number ‘n’ and print the sum of all number from 1 to n.**

**Ans: - Algorithm**

Step1: -start

Step2: - declare variable n, i, sum

Step3: - initialize i=1, sum=0

Step4: - read n

Step5: -if i<=n, the the number till n will keep adding , else the loop will end.

Step6: -display sum

Step7: -stop

**Pseudocode.**

Int main()

Printf(“enter a value”)

n, i=1, sum=0

if(i<=n)

Sum=sum+i

printf(“%d”,sum)

**Flowchart.**

start

declare variable n, i, sum

initialize i=1, sum=0

Read ‘n’

i=1

NO

YES

i<=n

Sum=sum+i

Print sum

start