Programming Using C
Week 04-1

Looping
while & do-while

Prepared By: REC Faculty 4.0 Team

LOOPING STATEMENTS

- Loops in programming come into use when we need to repeatedly execute a block of statements.

- For example: Suppose we want to print “Hello World” 10 times.

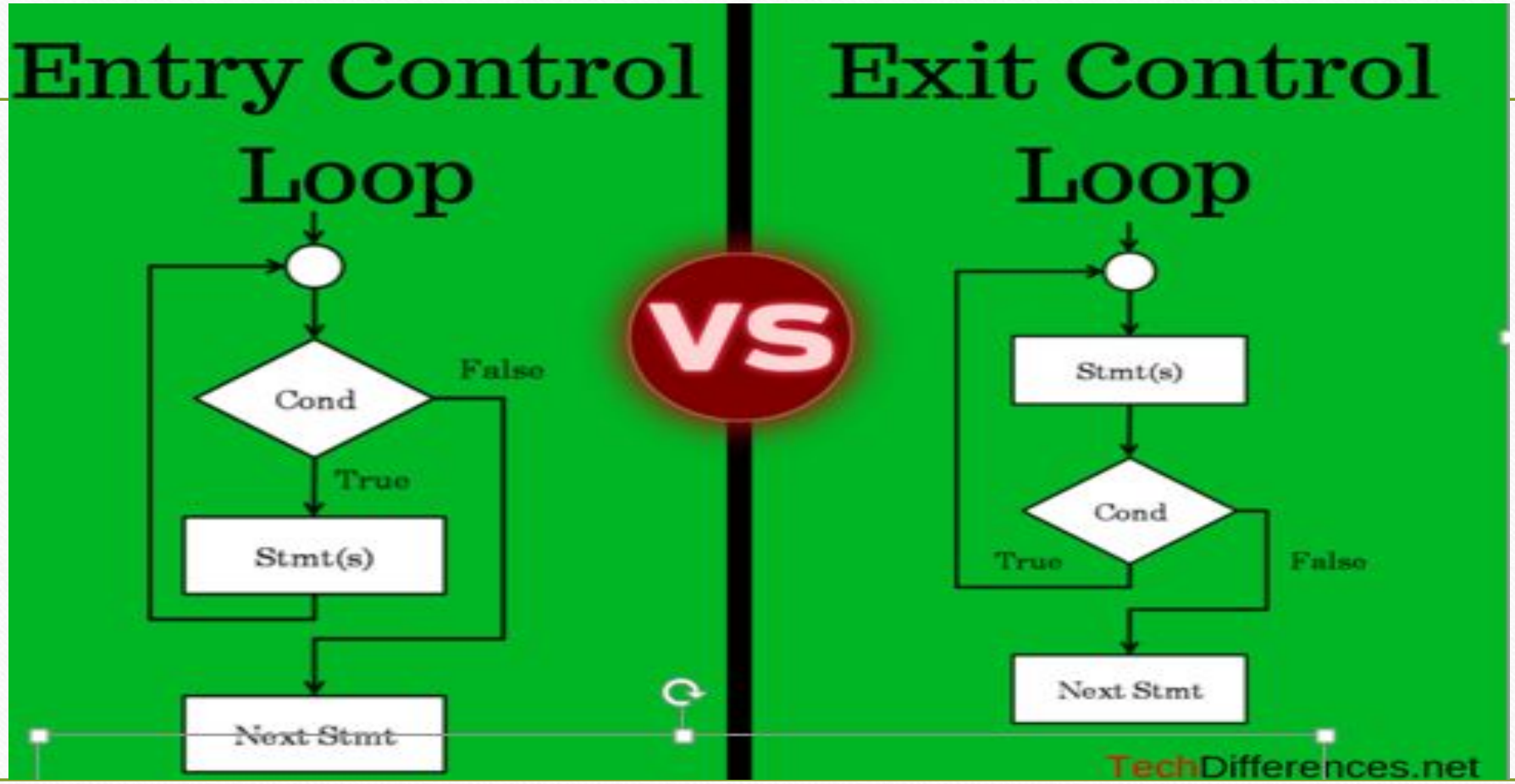
- DEFINITION:A loop is a sequence of instructions that is repeated until a certain condition is reached.

LOOPING STATEMENTS-TYPES

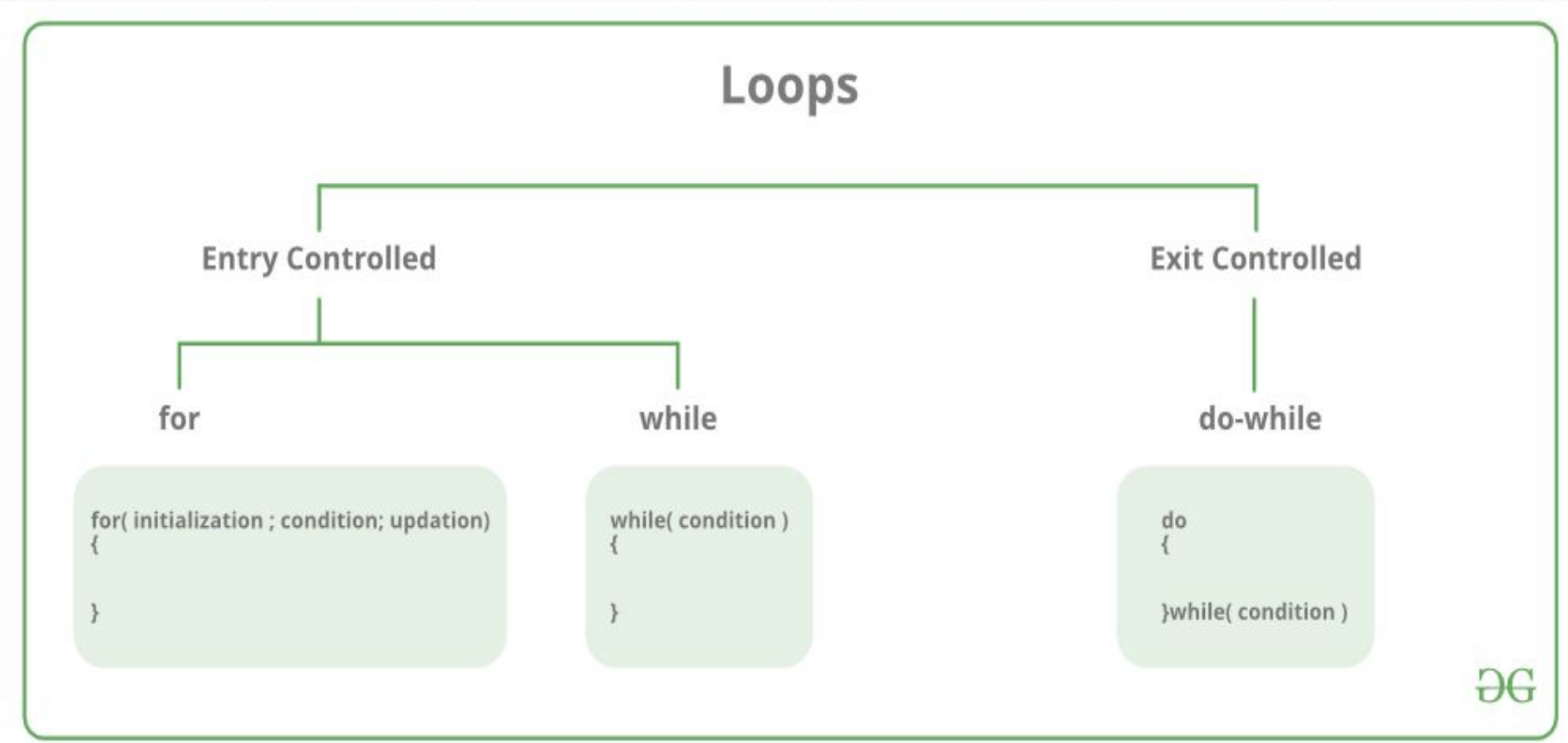
Entry Controlled

Exit Controlled

LOOPING STATEMENTS-TYPES



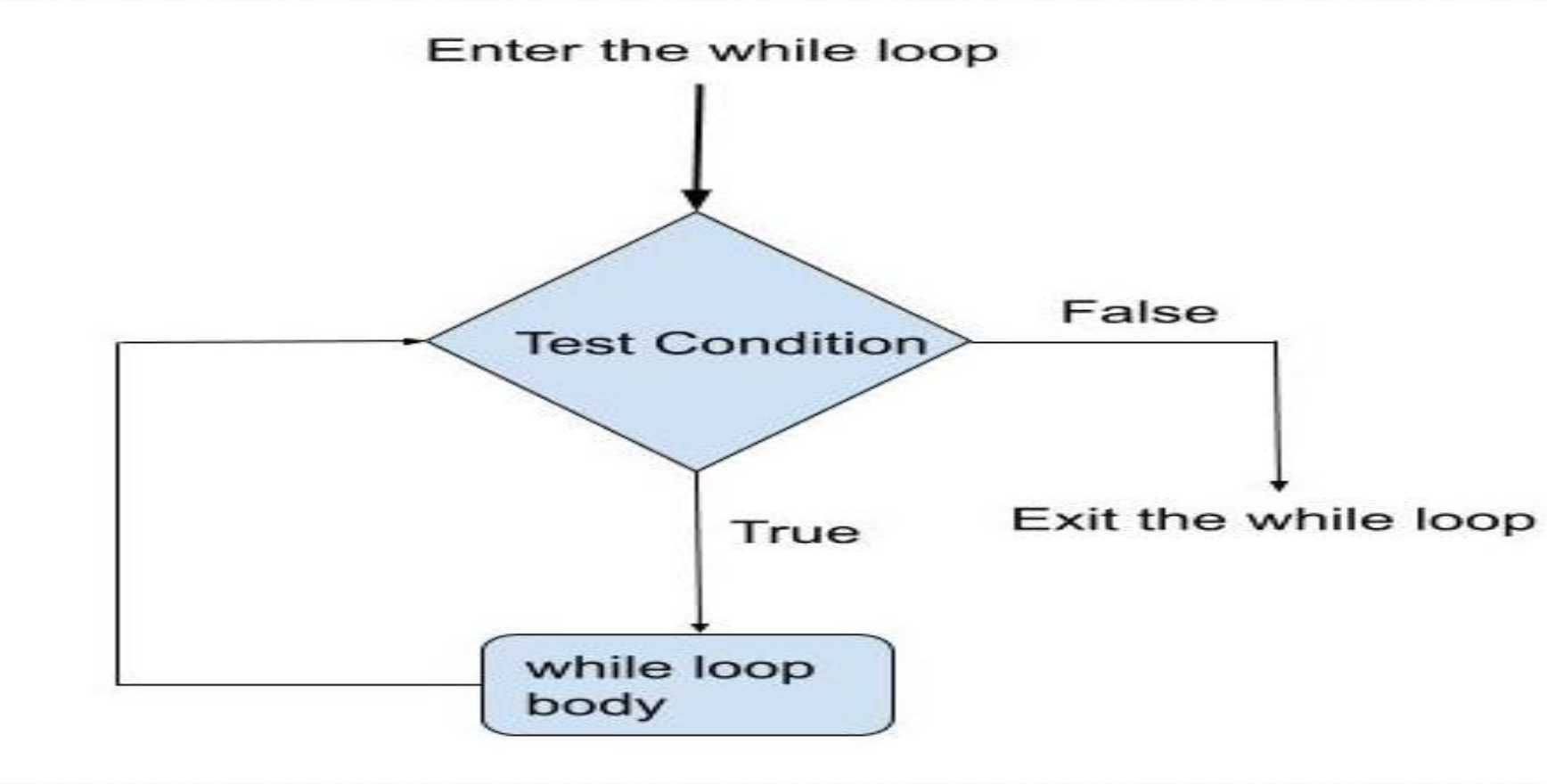
LOOPING STATEMENTS-TYPES



- A while loop is **a control flow statement that allows code to be executed repeatedly based on a given Boolean condition.**
- The while loop can be thought of as a **repeating if statement.**
- While loop has the following syntax:

```
while ( condition )
{
        statements;
}
```

While Loop


Enter the while loop
Test Condition
False
True
Exit the while loop
while loop
body

while Loop

- An example of a while statement:

```
int count = 0;
while (count < 2)
{
    printf("Welcome to C!");
    count++;
}
```

- If the condition of a while loop is false initially, the statement is never executed
- Therefore, the body of a while loop will execute zero or more times

Trace while Loop

```
int count = 0;
while (count < 2)
{
    printf("Welcome to C!");
    count++;
}
```

Initialize count

Trace while Loop, cont.

(count < 2) is true

```
int count = 0;
while (count < 2)
{
    printf("Welcome to C!");
    count++;
}
```

Trace while Loop, cont.

Print Welcome to C

```
int count = 0;
while (count < 2)
{
  printf("Welcome to C!");
  count++;
}
```

Trace while Loop, cont.

```
int count = 0;
while (count < 2)
{
  printf("Welcome to C!");
  count++;
}
```

Increase count by 1
count is 1 now

Trace while Loop, cont.

```
int count = 0;
while (count < 2)
{
    printf("Welcome to C!");
    count++;
}
```

(count < 2) is still true since count is 1

Trace while Loop, cont.

```
int count = 0;
while (count < 2)
{
    printf("Welcome to C!");
    count++;
}
```

Print Welcome to C

Trace while Loop, cont.

```
int count = 0;
while (count < 2)
{
    printf("Welcome to C!");
    count++;
}
```

Increase count by 1
count is 2 now

Trace while Loop, cont.

```
int count = 0;
while (count < 2)
{
    printf("Welcome to C!");
   count++;
}
```

(count < 2) is false since count is 2 now

Trace while Loop

```
int count = 0;
while (count < 2)
{
    printf("Welcome to C!");
   count++;
}
```

The loop exits. Execute the next statement after the loop.

Print numbers from 1 to 5

```
#include <stdio.h>
int main()
{
   int  i = 1;
   while (i <= 5)
    {
         printf("%d", i);
         ++i;
    }
   return 0;
}
```

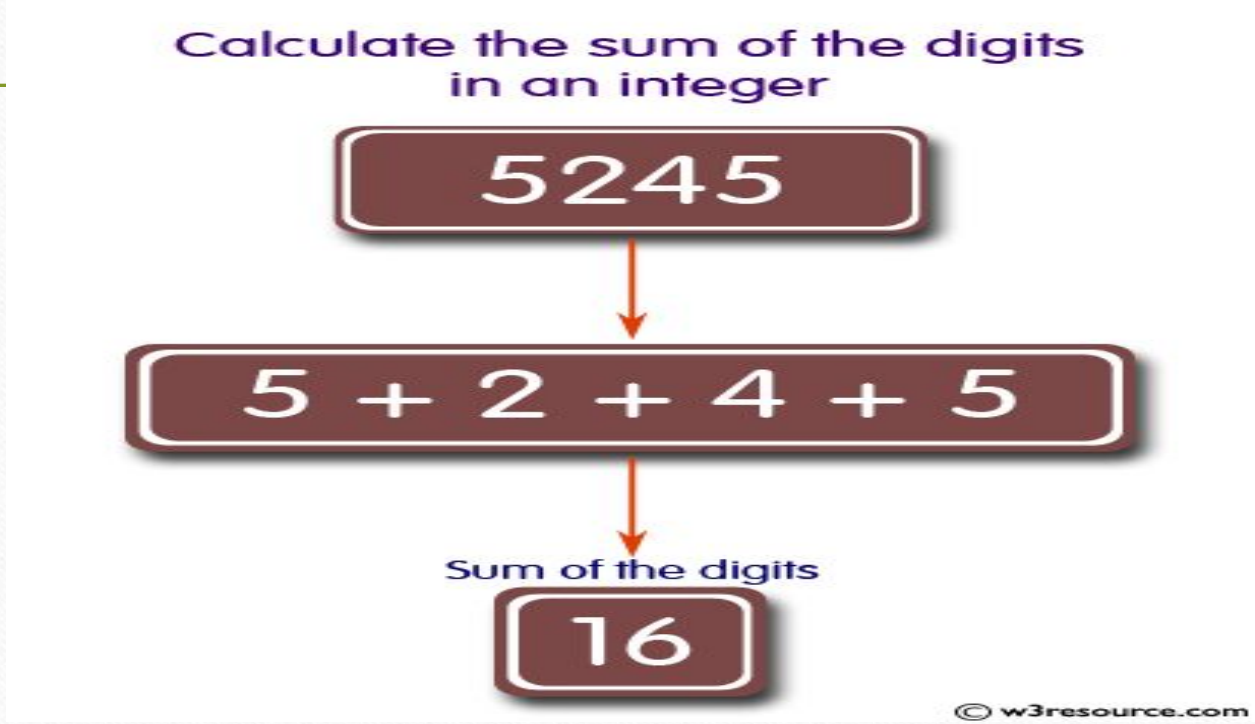
Output: 1 2 3 4 5

Program to Count the Number of Digits

```c
#include <stdio.h>
int main()
 {
   long long n;
   int count = 0;
   printf("Enter an integer: ");
   scanf("%lld", &n);
   while (n != 0) {
         n =n/10;
         ++count;
    }
    printf("Number of digits: %d", count);
}
```

```
int n=523
chk codn-(n!=0)-true
    n=523/10=52.3=52->incr cnt=1
chk codn-(n!=0)-true
    n=52/10=5.2=5->incr cnt=2
chk codn-(n!=0)-true
    n=5/10=0.5=0->incr cnt=3
chk codn-(n!=0)-false
```

??????


Calculate the sum of the digits in an integer
5245
5 + 2 + 4 + 5
Sum of the digits
16
© w3resource.com

Reverse Given Number

1234 → 4321

```c
#include <stdio.h>
int main() {
    int n, rev = 0, remainder;
    printf("Enter an integer: ");
    scanf("%d", &n);
    while (n != 0) {
        remainder = n % 10;
        rev = rev * 10 + remainder;
        n /= 10;
    }
    printf("Reversed number = %d", rev);
    return 0;
}
```

Perfect Number

Write a C program to check whether a given number is a perfect number or not. (Perfect number is a positive number which sum of all positive divisors excluding that number is equal to that number.
For example, 6 is perfect number since divisor of 6 are 1, 2 and 3.
Sum of its divisor is 1 + 2+ 3 = 6

```
/* perfect number */
#include<stdio.h>
int main()
{
   int num;
   int i=0;
   int sum=0;
   scanf("%d",&num);
   for(i=1;i<num;i++)
   {
      if(num%i==0)
        sum = sum + i;
   }
   if(sum==num)
       printf("YES");
   else
        printf("NO");
}
```

Sample input 1

6

YES

Sample input 2

10

NO

Infinite Loops

Infinite loops– test condition will be always true

Infinite Loops

- An example of an infinite loop:

```
int count = 1;
while (count <= 25)
{
    printf("%d",count);
    count = count - 1;
}
```

- This loop will continue executing until the user externally interrupts the program.

Nested Loops

- Similar to nested if statements, loops can be nested as well
- That is, the body of a loop can contain another loop
- For each iteration of the outer loop, the inner loop iterates completely

Nested Loops

- How many times will the string "Here" be printed?

```
count1 = 1;
while (count1 <= 10)
{
    count2 = 1;
    while (count2 <= 20)
    {
        printf ("Here");
        count2++;
    }
    count1++;
}
```

10 * 20 = 200

Stone Game-One Four

Alice and Bob are playing a game called "Stone Game". Stone game is a two-player game. Let N be the total number of stones. In each turn, a player can remove either one stone or four stones. The player who picks the last stone, wins. They follow the "Ladies First" norm. Hence Alice is always the one to make the first move. Your task is to find out whether Alice can win, if both play the game optimally.

Input Format

First line starts with T, which is the number of test cases. Each test case will contain N number of stones.

Output Format

Print "Yes" in the case Alice wins, else print "No".

Constraints

1<=T<=1000

1<=N<=10000

Sample Input and Output

Input

3
1
6
7

Output

Yes
Yes
No

Holes in a Number

You are designing a poster which prints out numbers with a unique style applied to each of them. The styling is based on the number of closed paths or holes present in a given number.

The number of holes that each of the digits from 0 to 9 have are equal to the number of closed paths in the digit. Their values are:

1, 2, 3, 5, and 7 = 0 holes.
0, 4, 6, and 9 = 1 hole.
8 = 2 holes.

Given a number, you must determine the sum of the number of holes for all of its digits. For example, the number 819 has 3 holes.

Complete the program, it must must return an integer denoting the total number of holes in num.

Constraints

$1 \leq num \leq 109$

Input Format For Custom Testing

There is one line of text containing a single integer num, the value to process.

Sample Input

630

Sample Output

2

Philaland Coin

The problem solvers have found a new Island for coding and named it as Philaland. These smart people were given a task to make a purchase of items at the Island easier by distributing various coins with different values. Manish has come up with a solution that if we make coins category starting from $1 till the maximum price of the item present on Island, then we can purchase any item easily. He added the following example to prove his point.

Let's suppose the maximum price of an item is 5$ then we can make coins of {$1, $2, $3, $4, $5}to purchase any item ranging from $1 till $5.

Now Manisha, being a keen observer suggested that we could actually minimize the number of coins required and gave following distribution {$1, $2, $3}. According to him any item can be purchased one time ranging from $1 to $5. Everyone was impressed with both of them. Your task is to help Manisha come up with a minimum number of denominations for any arbitrary max price in Philaland.

Input Format

Contains an integer N denoting the maximum price of the item present on Philaland.

Output Format

Print a single line denoting the minimum number of denominations of coins required.

Constraints

1<=T<=100
1<=N<=5000

Refer the sample output for formatting

Sample Input 1:

10

Sample Output 1:

4

MCQ

What will be the output of the C program?

```c
#include <stdio.h>
int main()
{
        int i = 5;
        while (--i > 0)
                printf("Loop ");
        return 0;
}
```

A. Loop Loop Loop Loop Loop Loop
B. Loop Loop Loop Loop Loop
C. Loop Loop Loop Loop
D. Loop Loop Loop

Answer C

What will be the output of the C program?

```c
#include <stdio.h>
int main()
{
        int i = 5, j = 0;
        while (i - j)
                printf("REC ");
        return 0;
}
```

A. REC REC REC REC REC
B. REC REC REC REC
C. REC REC
D. Infinite Loop

Answer D

```c
#include <stdio.h>
int main()
{
	int i = 0;
	while (i == 1)
	{
		printf("inside loop");
	}
	printf("outside loop");
	return 0;
}
```

A. outside loop
B. inside loop
C. inside loop outside loop
D. None of the above

Answer : A

What will be the output after execution of the program?

```
#include <stdio.h>
int main()
{
      int i = 5;
      while (i)
            printf("Hello");
      printf("World");
      return 0;
}
```

A. No output
B. Infinite time print of 'Hello'.
C. Infinite time print of 'World'.
D. Infinite time print of 'HelloWorld'.

Answer : B

Compute the printed value of i of the C program given below:

```
#include <stdio.h>
int main()
{
      int i = 0, j = 0;
      while (i < 4, j < 5)
      {
            i++;
            j++;
      }
      printf("%d, %d\n", i, j);
      return 0;
}
```

A. 4, 5
B. 4, 4
C. 5, 5
D. 0, 0

ANSWER: C

The while condition checks the last condition (i.e. j<5) and till the condition is satisfied the block inside the loop is executed. Thus the loop runs for 5 times and both the values of i and j are incremented by 5.

What will be the output of the C program?

```
#include <stdio.h>
int main()
{
        while (printf("%d", 5) < 4)
                printf("Loop ");
        return 0;
}
```

A. Prints Nothing
B. 5Loop 5Loop 5Loop 5Loop 5Loop
C. 5Loop
D. Infinite iterations or Infinite Loop

ANSWER: D

Explanation
We don't have any control in the above C program because there is no variables. If there is any variable we can control it. The condition is (printf("%d", 5) < 4) it will be (1 < 4), the condition is always true. So the loops is infinite. The same program using some variable.

What will be the output of the C program?

```c
#include <stdio.h>
int main()
{
	int i = 0;
	while (0, i < 4)
	{
		printf("Loop ");
		i++;
	}
	return 0;
}
```

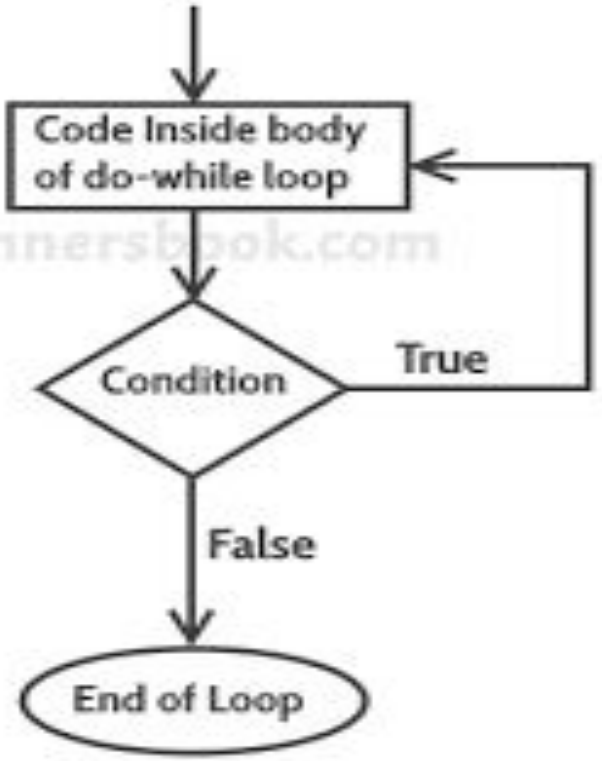
A. Prints Nothing
B. Infinit Loop
C. Loop Loop Loop Loop
D. Loop Loop Loop Loop Loop

ANSWER: C

do..while loop

- The do..while loop is similar to the while loop with one important difference. The body of do...while loop is executed at least once.
- Only then, the test expression is evaluated.

LOOPING STATEMENTS-do while



SYNTAX:

```
do
 {
        //Statements
  }while(condition test);
```

While vs Do-While Loop

Comparison Chart

While Loop	Do-While Loop
The while loop evaluates the condition first and then execute the statements.	The do-while loop executes the statements first before evaluating the condition.
The condition is specified at the beginning of the loop.	The condition isn't specified until after the body of the loop.
The body is executed only if a certain condition is met and it terminates when the condition is false.	The body is always executed at least once, regardless of whether the condition is met.

LOOPING STATEMENTS-programming logics

- Write a program to count the number of spaces between numbers.(consider that their will only one space between two numbers)
- Input: 23 45 34 56 72
- Output: 4

LOOPING STATEMENTS-programming logics

```c
#include<stdio.h>
int main()
{
    int a;
    char ch;
    int count=-1;
    do
    {
        count++;
        scanf("%d",&a);
        scanf("%c",&ch);
    }while(ch!='\n');
printf("The no of spaces is: %d",count);
}
```

LOOPING STATEMENTS-programming logics

- Write a program to count the odd numbers ,given a set of numbers separated by space in a line .(consider that their will only one space between two numbers)
- Input: 23 45 34 56 72
- Output:2

LOOPING
STATEMENTS-programming logics

```c
#include<stdio.h>
int main()
{
    int a;
    char ch;
    int count=0;
    do
    {
        scanf("%d",&a);
        if(a%2==1)
            count++;
        scanf("%c",&ch);
    }while(ch!='\n');
printf("The no of odds is: %d",count);
}
```

MCQ

What is the output of the following program?

```
#include <stdio.h>
int main()
{
        int i = 1;
        do
        {
                printf("%d ", 5 * i);
                i++;
        } while (i < 4);
        return 0;
}
```

A. 5 10 15
B. 5 10 15 20
C. 10 15 20
D. Prints Nothing

Answer A

What will be the output of the program?

```
#include <stdio.h>
int main()
{
      int x = 25, y = 1;
      do
            if (x > 5)
                  printf(" ONE");
            else if (x > 10)
                  printf(" TWO");
            else if (x == 25)
                  printf(" THREE");
            else
                  printf(" FOUR");
      while (y--);
      return 0;
}
```

A. ONE ONE
B. ONE TWO THREE
C. THREE
D. Compile time error

ANSWER: A

What will be the output of following program?

```
#include <stdio.h>
int main()
{
	int cnt = 1;
	do
	{

		printf("%d, ", cnt);
		cnt += 1;
	} while (cnt >= 10);
	printf("After loop cnt=%d", cnt);
	return 0;
}
```

A. After loop cnt=1
B. 1, After loop cnt=2
C. After loop cnt=2
D. compile time error

ANSWER: B

Find the output:

```
#include <stdio.h>
int main()
{
	int i = 0;
	do
	{
		printf("inside loop ");
	} while (i == 1);
	printf("outside loop");
	return 0;
}
```

A. inside loop
B. outside loop
C. inside loop outside loop
D. None of the above

ANSWER: C

How many times i value is checked in the following C code?

```c
#include <stdio.h>
int main()
{
	int i = 0;
	do
	{
		i++;
		printf("in while loop\n");
	} while (i < 3);
	return 0;
}
```

A. 2
B. 3
C. 4
D. 1

ANSWER: B

What will be the output of the following C code?

```c
#include <stdio.h>
int main()
{
	int i = 2;
	do
	{
		printf("Hi ");
	} while (i < 2)
	return 0;
}
```

A. Compile time error
B. Hi Hi
C; Hi
D. Varies

ANSWER: A

What is the output of C Program?

```c
#include <stdio.h>
int main()
{
	int a = 32;
	do
	{
		printf("%d", a);
		a++;
	} while (a <= 30);
	return 0;
}
```

A. 32
B. 33
C. 30
D. No Output

ANSWER: A


Thank You