23MX128

Lt Col T V Perumal(Retd)

**Odd Even strategy**

Declare a single array of size N elements

Divide the array into 2 logical arrays to represent the two stacks

Name the two stacks as EvenStack and OddStack

EvenStack holds data at even indices of the array i.e. 0, 2, 4, 6, ….and so on

OddStack holds data at odd indices of the array i.e. 1, 3, 5, 7,…. and so on

Declare variables as under

OddStackTop – indicates the index of the topmost element in OddStack

EvenStackTop – indicates the index of the topmost element in EvenStack

OddStackSize – the number of elements in OddStack

EvenStackSize – the number of elements in EvenStack

Initialize variables as under

OddStackTop = -1; OddStackSize = 0;

EvenStacTop = -2; EvenStackSize = 0;

1. To push element into stacks :-

If N is an odd number

{ If OddStackTop < (N-2) push into OddStack, call Push function

Else if EvenStackTop < (N-1) push into EvenStack, call Push function

Else print “Stacks are full”

}

If N is even number

{ If OddStackTop < (N-1) push into OddStack, call Push function

Else if EvenStackTop < (N-2) push into EvenStack, call Push function

Else print “Stacks are full”

}

Function Push (Stack, X)

{ If Stack passed is OddStack

{ OddStackTop +=2

OddStack[OddStackTop] = X

}

Else // Stack passed is EvenStack

{ EvenStackTop +=2

EvenStack[EvenStackTop] = X

}

}

2. To pop elements from respective stacks

Function Pop (Stack)

{ If Stack passed is OddStack

{ If OddStackTop == -1 print “Stack Empty” return

Else { Print “Element popped : “ + OddStack[OddStackTop]

OddStackTop -=2

Call SizeOfStack function to update size of OddStack

}

}

Else // Stack passed is EvenStack

{ If EventStackTop == -2 print “Stack Empty” return

Else { Print “Element popped : “ + EvenStack[EvenStackTop]

EvenStackTop -=2

Call SizeOfStack function to update EvenStack

}

}

}

3. Function SizeOfStack(Stack)

{ if OddStack

{ if OddStackTop == -1 return 0

Else { OddStackSize = OddStackTop/2 + 1

Return OddStackSize

}

}

Else // its EvenStack

{ if EvenStackTop == -1 return 0

Else { EvenStackSize = EvenStackTop/2 + 1

Return EvenStackSize

}

}

}