28/9/21

DS

Date
Page 1

Prajapati Yash. P. 201621210 23 (BDA)

(1) Given :-

Arr1	11	12	13	14	
Arr2	21	22	23	24	25

Req. output :-

Result	21	22	23	24	25	14	13	12	11

Sol^n Algorithm :-

Step 1 :- we will set i=0, j = length(Arr1) - 1
Step 2 :- while i<j repeat steps 3 to 5
Step 3 :- swap (Arr1[i], Arr[j])
Step 4 :- i++
Step 5 :- j--
Step 6 :- Exit
Step 7 :- Again set i=0 & j=0
Step 8 :- while Arr2[i] != NULL repeat step 9
Step 9 = i++
Step 10 :- while rev- Arr1[j] = NULL repeat step 11 to 13
Step 11 :- Arr2[i] = rev- Arr1[j]
Step 12 :- i++
Step 13 :- j++
Step 14 :- Arr2[i] = NULL
Step 15 :- Exit

C-code :-

```
#include <stdio.h>

int main()
{
```

Prajapati Yash.P. 201621210 23 (BDA)

```
int Arr1[] = {11, 12, 13, 14};
int Arr2[] = {21, 22, 23, 24, 25};
int i, j = 0;
int rev_Arr1[10];
int count;

while (Arr1[count] != '\0')
  {
    count++;
  }
j = count - 1;

while (i<j) {
    rev_Arr1[i] = Arr1[j];
    j--;
    i++;
  }
rev_Arr1[i] = '\0';

i=0;

while (Arr2[i] != '\0')
  {
    i++;
  }
g=0;
while (rev_Arr1[j] != '\0')
{
    Arr2[i] = rev_Arr1[j];
```

```
        i++;
        j++;
    }
    Arr2[i] = '\0';
    for(i=0; i<strlen
    printf("Result: %d",
    printf("Result: \n");
    for (j=0 ; j<i ; j++)
    {
        printf (" %d ", Arr2[j]);
    }

    return 0;
}
```

Q3 Given :- str = "Hypothetically"

Algo :- (i) set i=0, j=length(str) - 1
(ii) while i<j repeat step 3 to 5
(iii) swap (str[i], str[j])
(iv) i = i+1
(v) j = j-1
(vi) Exit.

Soln # Code :-

```
#include <stdio.h>

int main()
{   char str[20];
```

Prajapati Yash .P. 201621210 23 (BDA)

```
int i=0, j, count;
char rstr[20];
while (str[count] != '\0')
{
    count ++;
}

j= count - 1;

while (i<j)
{
    rstr[i] = str[j];
    j--; i++;
}
rstr[i]='\0';

printf("Result = %s ", rstr);

return 0;
}
```

Iterations :-

Consider str given = "Hypothetically"

⇒ After first while loop,

i = 0

count = 14

j = 13

Prajapati Yash.P. 201621210 23 (BDA)

⇒ In next while loop,

Iteration 1 :-

i = 0 , j = 13

str[0] = str[13]

∴ str[0] = str[13] = y

Iteration 2 :-

i = 1 , j = 12

∴ str[1] = str[12] = l

Iteration 3 :-

i = 2 , j = 11

∴ str[2] = str[11] = l

∴ output upto 3 iteration = yll

And, the original output = yllacitehtopyH

Prajapati Yash.P. 201621210 23 (BDA)

Q.5 Infix expression A+B* (C^D)-(E/F)

Sol^n Algorithm :-

- Step 1 :- Scan the infix Expression from left to right.

Step 2 :- If the scanned is operand, show it in output.

Step 3 :- Else,

↳ Step 3.1 :- If the precedence of scanned op. is greater than precedance of op in stack, pust it.

↳ Step 3.2 :- Else, pop all the operators from stack which are greater than or equal to in precedence. After doing that, push the scanned operator to the stack.

Step 4 :- If the scanned char is '(', push it to stack.

Step 5 :- If the scanned char is ')', pop the stack & output it until a '(' is in stack and discard both parenthesis.

Step 6 :- Repeat step 2-6 until infix expression is scanned.

Step 7 :- Print output

Step 8 :- pop and output elements from stack until it is not empty.

For eg :-

Given Infix expression :- A + B * (C % D) - (E / F)

Output :- ABCD % * + EF / -

Stack :-

-	ABCD % * + EF / -
/	ABCD % * + EF /
F	ABCD % * + EF
E	ABCD % * + E
+	ABCD % * +
*	ABCD * % *
%	ABCD %
D	ABCD
C	ABC
B	AB
A	A