Peactical No 8

Aim:- write a c program for selection sort Generate equivalent 3add-code.

Tools :

SR.NO	Tools	specification	Qty
1	C compiler	Turbo C, all, any -	-
2	Computer system	windows, 4GB RAM -	1

Theory :-

Selection sort :- It's a simple and easy to understand sorting algorithm that works by repeatedly selecting the smallest element from unsorted pattern of the list.

It's repeated for the remaining unsorted ~~pattern~~ ~~of~~ portion of the list untile the entire list is sorted.
Three address code · It's a type of intermediate code which is easy to generates and can be easily converted to machine code it makes use of atmost three address code and one operator to represent an expression

Program :-

```
#include<stdio.h>
void selectionsort (int arr[], int n)
{
    int i, j, min_idx;
    for(i=0; i<n-1; i++)
    {
        min_idx=i;
        for (j=i+1 ; j<n ; j++)
          {
            if (arr[i] < arr[min-idx])
                 min_idx =j ;

            int  temp = arr [min_idx] ;
            arr[min_idx] == arr[i] ;
               arr[i] = temp ;
          }
     }
  int main() {
                  int arr[] = { 64, 25, 12, 22, 11 } ;
          int n = sizeof (arr) / size-of (arr[0]) ;
            selectionsort (arr, n) ;
           printf( "sorted  Array") ;
        for (int  i=0 ; i<n ; i++)
           printf("%d", arr[i]) ;
           return 0 ;
   }
```

3-address Code :-

```
                      Initialize variable
                           i = 0 ;
                           j = 0 ;
                      min_idx = 0 ;
                      ; start outer loop
                      L1 if  i >= n-1 go 16

                      ; start inner loop
                      L2 if j > n goto 14
```

```
if arr[j] < arr[min_idx] goto 13
   goto 14

; update min_idx
    L3 : min_idx = j
    L4 : j = j+1
    goto 12
    ; End inner loop

; swap arr[min_idx] with arr[i]
temp = arr[min_idx]
 arr [min idx] = arr [i]
     arr[i] = temp

     i = i+1
     j = i
     goto L1
  ; END outer loop
; print sorted array
   L5 : printf("sorted Array : ").
     for i=0 to n-1
       printf(" %d ", arr[i])
```

[

ERRORS and Remedial action :

ERRORS : Not handling dep

Remedial Action :- we are modify algorithm to keep touch of all occurences of the minimum element.

Conclusion : C program for selection sort and its equivalent 3 add code has been generate successfully.

Cource outcome attained & mapping with Program outcomes

Cource outcome	Program outcome
co-3 Apply the knowledge YACC to syntax divided translation for generating intermediate code 3-add code	PO2 : Problem Analysis PO3 : Design / Dev. pt of solution.

Peactical NO 09

Aim :- Write a Peogeam in C ~~peamo~~ program to ~~a~~ Optimize the 3-address code generated.

Tools :-

SE. NO	Toole	specification	Qty
1	compilee	~~G~~ computee (any)	-
		Turbo C ~~++~~ C++	
		of GCC	
2	computer system	4GB RAM	1
		windows, 1TB	
		HDD	

Theory :- 3Address code : It is a type of intermediate code which is easy to generate and can be easily converted to Machine code. It make use of at most 3address and one opeeatoe to represent an expression and the value computed at each instruction is stored in temporary variable generated by compilee.

3 Address code

Optimization :- 3Address Code is often used as an intermediate representation of code during optimization phases of the comilation process. the 3address code allows the compilee to analyze the and perform Optimization that can improvs the performance of the generated code.

PROGRAM :

Optimized Theee Address code :

```
1)  i=0
2)  j=0
3)  min_indx
4)  if i>n-1  goto (23)
5]  if j>=n  goto (13)
6)  t1 = arr[j]
7)  t2 = arr[min_ind]
8)  if t1<t2 , goto (10)
9)  goto (13)
10)  min_indxj ;
11)  t3= j+1 ;
12]  j= t3
13)  goto
14]   swap t2 with arr[i]
15]   t4 = arr[i]
16]   t5 = t2
17)   t2 = t4
18)   t4 = t5
19)    t6 = i+1
20)    i = t6
21)    j = i
22]   goto ()
23)   printf("sorted Array");
24)  for i=0 to n-1 goto (25]
25]  printf("%d", t4)
26]  goto (24)
```

~~ERROP~~ ERROR and Remedial Action

Error :- Not handling duplicate elements correctly

Remedial : We can modify algorithm to keep track of all occurence of the minimum element and sutup them all to correct position.

Conclusion :-

Program to implement C program for Selection sort, then generating three address code for it, and after this optimizing the three address code and generation of object code implemented Successfully.

Course outcome attained and Mapping with po's :

Course outcome	Program outcomes
CO-4 : Build a code generation using different intermediate ~~and~~ code and optimize the target code.	PO1 : Engineering knowledge PO2 : problem Analyze PO3 : Design development of solution PO5 : Modern tool usage PO6 : individual and team work.

PEactical No 10

Aim:- write a c peogram Foe selection sort generate and generator objeets code foe three address code.

Tools :-

SR.NO	Tools	specification	coby
1.	compilee	Tubbo c, c++ /GCC	-
2.	computer system	windows, 4GB RAM 1TB HDD	1

Theory :-

Three address code is on intermediate code it is used by the optimizing compilers

In 3-address code the given expression is beoken down into serveal separate insteuctions These insteuction can easily translate into assembly language such three address code inteuction has at most three operands. It is a combination of assigment and a binary operatoe

PROGRAM :- object code Generated feom optimized 3-address code :

```
1) start :
2) mov i, 0
3) mov j, 0
4) loop outee
5] compl n, i, imp loop
6] cmple n, j, imp (11)
7) mov aeeEij], t1
```

```
8]  mov t2 , arr[mid-ind]
9]  comp  t1 , t2
10) mov  min-ind , j
11) add j , 1
12) mov j , t3
13) mov t4 , arr[j]
14) mov t5 , t2
15) mov t2 , . t4
16) mov t4 , t5 , add j , 1
17) mov t6 , j
18) mov i , t6
19) mov j , i
20) loop end
21) loop
22)     jmp loop
```

Conclusion : Program to implement C program for selection sort then generating 3-address code for it, and after this optimization the 3-address code and generation of object code implemented successfully.