**IT300 – Design and Analysis of Algorithms**

Lab Assignment – 6

**Name:** Niraj Nandish

**Roll No:** 191IT234

1. Program 1 – Minimum number of platforms required
   1. Code

#include <iostream>

#include <stdio.h>

#include <vector>

using namespace std;

int minPlatforms(vector<int> *arrival*, vector<int> *departure*, int *n*)

{

int plat = 1, result = 1, i = 1, j = 0;

sort(*arrival*.begin(), *arrival*.end());

sort(*departure*.begin(), *departure*.end());

while (i < *n* && j < *n*)

{

if (*arrival*[i] <= *departure*[j])

{

plat++;

i++;

}

else

{

plat--;

j++;

}

result = max(result, plat);

}

return result;

}

int main()

{

int n, platforms;

cout << "Enter number of inputs of arrival and departure times: ";

cin >> n;

vector<int> arrival(n);

vector<int> departure(n);

cout << "\nEnter the time in 24hr format without ':' between hour and minute group.";

for (int i = 0; i < n; i++)

{

cout << "\n\nEnter arrival time for train " << i + 1 << ": ";

cin >> arrival[i];

cout << "Enter departure time for train " << i + 1 << ": ";

cin >> departure[i];

}

platforms = minPlatforms(arrival, departure, n);

cout << "\nMinimum number of platforms required: " << platforms << endl;

return 0;

}

* 1. Text

     Description automatically generatedScreenshots

1. Program 2 – Greedy Algorithms: Interval Scheduling
   1. Code

#include <iostream>

#include <stdio.h>

#include <vector>

using namespace std;

struct *performance*

{

int s;

int f;

};

bool compFunc(*performance* *m1*, *performance* *m2*)

{

return *m1*.f < *m2*.f;

}

void interval\_scheduling(vector<*performance*> *arr*, int *n*)

{

int pos = 0;

sort(*arr*.begin(), *arr*.end(), compFunc);

cout << "\nThe following intervals are selected : ";

cout << "(" << *arr*[0].s << ", " << *arr*[0].f << ") ";

for (int i = 1; i < *n*; i++)

{

if (*arr*[i].s >= *arr*[pos].f)

{

cout << "(" << *arr*[i].s << ", " << *arr*[i].f << ") ";

pos = i;

}

}

}

int main()

{

int n;

cout << "Enter number of intervals: ";

cin >> n;

vector<*performance*> arr(n);

for (int i = 0; i < n; i++)

{

cout << "\nEnter the start time of interval " << i + 1 << ": ";

cin >> arr[i].s;

cout << "Enter the finish time of interval " << i + 1 << ": ";

cin >> arr[i].f;

}

interval\_scheduling(arr, n);

cout << endl;

return 0;

}

* 1. Text

     Description automatically generatedScreenshots