SOFTWARE PROJECT MANAGEMENT LAB - G2
EXPERIMENT 10

- ASHISH KUMAR
- 2K18/SE/041

AIM:- Write a program to implement class point method.

THEORY:-

- Class point method is used to provide system level size estimation of object oriented software.
- This method was given by Gennaro Costagliola et al. in 2005.
- This method primarily focuses on classes for the estimation of size.
- The final class point is calculated by multiplying total unadjusted class point values with technical factor:

  CP = TUCP x TCF

CODE:-

```
#include<bits/stdc++.h>
#include<iostream>
using namespace std;

int w[4][3];
string sys_char[18] = {
  "Data Communication",
  "Distributed Functions",
  "Performance",
  "Heavily used configuration",
  "Transaction rate",
  "Online data entry",
  "End-user efficiency",
  "Online Update",
  "Complex Processing",
  "Re-usability",
  "Installation ease",
  "Operational ease",
  "Multiple sites",
  "Facilitation of change",
  "User adaptivity",
```

```
  "Rapid prototyping",
  "Multiuser Interactivity",
  "Multiple Interfaces"
};

float cal_cp1(int nem, int nsr, int nda, int ctype) {

  if (nsr < 0 or nem < 0 or ctype < 0)
    return -1;

  int compxty = 0;
  if (nem >= 0 and nem <= 4) {
    if (nsr >= 0 and nsr <= 3) {
      compxty = 0;
    } else if (nsr >= 4) {
      compxty = 1;
    }
  } else if (nem >= 5 and nem <= 8) {
    if (nsr >= 0 and nsr <= 1) {
      compxty = 0;
    } else if (nsr >= 2 and nsr <= 3) {
      compxty = 1;
    } else if (nsr >= 4) {
      compxty = 2;
    }
  } else if (nem >= 9) {
    if (nsr >= 0 and nsr <= 1) {
      compxty = 1;
    } else if (nsr >= 2) {
      compxty = 2;
    }
  } else {
    return -1;
  }

  return w[ctype][compxty];
}

float cal_cp2(int nem, int nsr, int nda, int ctype) {

  if (nda < 0 or nsr < 0 or nem < 0 or ctype < 0)
    return -1;

  int compxty = 0;
  if (nsr >= 0 and nsr <= 2) {
    if (nda >= 0 and nda <= 5) {
```

```
    if (nem >= 0 and nem <= 8) {
      compxty = 0;
    } else {
      compxty = 1;
    }
  } else if (nda >= 6 and nda <= 9) {
    if (nem >= 0 and nem <= 4) {
      compxty = 0;
    } else if (nem >= 5 and nem <= 8) {
      compxty = 1;
    } else {
      compxty = 2;
    }
  } else if (nda >= 10) {
    if (nem >= 0 and nem <= 4) {
      compxty = 1;
    } else {
      compxty = 2;
    }
  }
} else if (nsr >= 3 and nsr <= 4) {
  if (nda >= 0 and nda <= 4) {
    if (nem >= 0 and nem <= 7) {
      compxty = 0;
    } else {
      compxty = 1;
    }
  } else if (nda >= 5 and nda <= 8) {
    if (nem >= 0 and nem <= 3) {
      compxty = 0;
    } else if (nem >= 4 and nem <= 7) {
      compxty = 1;
    } else {
      compxty = 2;
    }
  } else if (nda >= 9) {
    if (nem >= 0 and nem <= 3) {
      compxty = 1;
    } else {
      compxty = 2;
    }
  }
} else if (nsr >= 5) {
  if (nda >= 0 and nda <= 3) {
    if (nem >= 0 and nem <= 6) {
      compxty = 0;
```

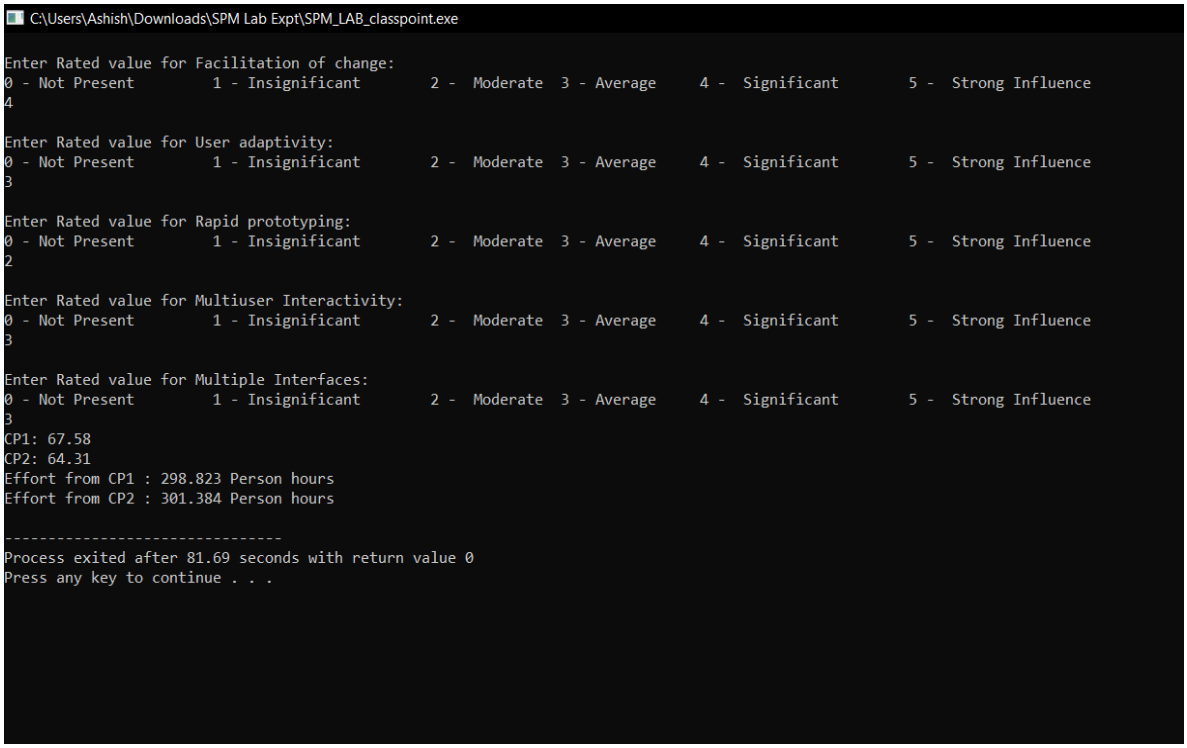
```
      } else {
        compxty = 1;
      }
    } else if (nda >= 4 and nda <= 7) {
      if (nem >= 0 and nem <= 2) {
        compxty = 0;
      } else if (nem >= 3 and nem <= 6) {
        compxty = 1;
      } else {
        compxty = 2;
      }
    } else if (nda >= 8) {
      if (nem >= 0 and nem <= 2) {
        compxty = 1;
      } else {
        compxty = 2;
      }
    }
  }
  return w[ctype][compxty];
}

int main() {

  w[0][0] = 3;
  w[0][1] = 6;
  w[0][2] = 10;

  w[1][0] = 4;
  w[1][1] = 7;
  w[1][2] = 12;

  w[2][0] = 5;
  w[2][1] = 8;
  w[2][2] = 13;

  w[3][0] = 4;
  w[3][1] = 6;
  w[3][2] = 9;

  int TUCP1 = 0, TUCP2 = 0;
  int i = 0;
  int nem, nsr, nda;
  cout << "---------- CLASS POINT METHOD ----------\n\n";
  while (1) {
```

```
    int ctype;
    cout << "Enter -1 or Choose ctype by: 0 - PDT\t1 - HIT\t2 - DMT\t3 - TMT and\n";
    cout << "Enter details separated by space | ctype nem, nsr and nda: ";
    cin >> ctype;

    if (ctype < 0)
      break;

    cin >> nem >> nsr >> nda;
    int v1 = cal_cp1(nem, nsr, nda, ctype);
    int v2 = cal_cp2(nem, nsr, nda, ctype);

    if (v1 < 0 or v2 < 0) {
      cout << "Wrong Entry | Enter Non-Negative numbers only\n\n";
      continue;
    }

    TUCP1 += v1;
    TUCP2 += v2;
    cout << endl;
  }

  float TDI = 0;
  for (int i = 0; i < 18; i++) {
    cout << "\nEnter Rated value for " << sys_char[i] <<":"<< endl;
    cout << "0 - Not Present\t\t1 - Insignificant\t 2 -  Moderate\t3 - Average\t4 -  Significant\t5 -
Strong Influence\n";
    int x;
    cin >> x;

    if (x < 0 or x > 5) {
      cout << "Enter a number in range 0->5\n";
      i--;
      continue;
    }
    TDI += x;
  }
  float TCF = 0.55 + 0.01 * TDI;
  float CP1 = TUCP1 * TCF;
  float CP2 = TUCP2 * TCF;
  cout << "CP1: " << CP1 << endl;
  cout << "CP2: " << CP2 << endl;
  cout << "Effort from CP1 : " << 0.843 * CP1 + 241.853 << " Person hours" << endl;
  cout << "Effort from CP2 : " << 0.912 * CP1 + 239.751 << " Person hours" << endl;
return 0;
}
```

OUTPUT:-

```
C:\Users\Ashish\Downloads\SPM Lab Expt\SPM_LAB_classpoint.exe
--------------- CLASS POINT METHOD ---------------

Enter -1 or Choose ctype by: 0 - PDT    1 - HIT 2 - DMT 3 - TMT and
Enter details separated by space | ctype nem, nsr and nda: 0 4 1 10

Enter -1 or Choose ctype by: 0 - PDT    1 - HIT 2 - DMT 3 - TMT and
Enter details separated by space | ctype nem, nsr and nda: 0 3 3 12

Enter -1 or Choose ctype by: 0 - PDT    1 - HIT 2 - DMT 3 - TMT and
Enter details separated by space | ctype nem, nsr and nda: 0 4 1 5

Enter -1 or Choose ctype by: 0 - PDT    1 - HIT 2 - DMT 3 - TMT and
Enter details separated by space | ctype nem, nsr and nda: 1 1 1 1

Enter -1 or Choose ctype by: 0 - PDT    1 - HIT 2 - DMT 3 - TMT and
Enter details separated by space | ctype nem, nsr and nda: 1 4 2 1

Enter -1 or Choose ctype by: 0 - PDT    1 - HIT 2 - DMT 3 - TMT and
Enter details separated by space | ctype nem, nsr and nda: 1 8 2 1

Enter -1 or Choose ctype by: 0 - PDT    1 - HIT 2 - DMT 3 - TMT and
Enter details separated by space | ctype nem, nsr and nda: 2 3 3 0

Enter -1 or Choose ctype by: 0 - PDT    1 - HIT 2 - DMT 3 - TMT and
Enter details separated by space | ctype nem, nsr and nda:  2 4 2 0

Enter -1 or Choose ctype by: 0 - PDT    1 - HIT 2 - DMT 3 - TMT and
Enter details separated by space | ctype nem, nsr and nda: 2 5 3 0

Enter -1 or Choose ctype by: 0 - PDT    1 - HIT 2 - DMT 3 - TMT and
Enter details separated by space | ctype nem, nsr and nda: 2 6 3 2

Enter -1 or Choose ctype by: 0 - PDT    1 - HIT 2 - DMT 3 - TMT and
Enter details separated by space | ctype nem, nsr and nda: 3 2 0 0

Enter -1 or Choose ctype by: 0 - PDT    1 - HIT 2 - DMT 3 - TMT and
Enter details separated by space | ctype nem, nsr and nda: 3 2 2 0

Enter -1 or Choose ctype by: 0 - PDT    1 - HIT 2 - DMT 3 - TMT and
Enter details separated by space | ctype nem, nsr and nda: 3 3 1 1

Enter -1 or Choose ctype by: 0 - PDT    1 - HIT 2 - DMT 3 - TMT and
Enter details separated by space | ctype nem, nsr and nda: -1

Enter Rated value for Data Communication:
0 - Not Present         1 - Insignificant       2 -  Moderate  3 - Average     4 -  Significant       5 -  Strong Influence
3
```

```
C:\Users\Ashish\Downloads\SPM Lab Expt\SPM_LAB_classpoint.exe

Enter Rated value for Distributed Functions:
0 - Not Present         1 - Insignificant       2 -  Moderate  3 - Average     4 -  Significant       5 -  Strong Influence
2

Enter Rated value for Performance:
0 - Not Present         1 - Insignificant       2 -  Moderate  3 - Average     4 -  Significant       5 -  Strong Influence
3

Enter Rated value for Heavily used configuration:
0 - Not Present         1 - Insignificant       2 -  Moderate  3 - Average     4 -  Significant       5 -  Strong Influence
3

Enter Rated value for Transaction rate:
0 - Not Present         1 - Insignificant       2 -  Moderate  3 - Average     4 -  Significant       5 -  Strong Influence
3

Enter Rated value for Online data entry:
0 - Not Present         1 - Insignificant       2 -  Moderate  3 - Average     4 -  Significant       5 -  Strong Influence
3

Enter Rated value for End-user efficiency:
0 - Not Present         1 - Insignificant       2 -  Moderate  3 - Average     4 -  Significant       5 -  Strong Influence
3

Enter Rated value for Online Update:
0 - Not Present         1 - Insignificant       2 -  Moderate  3 - Average     4 -  Significant       5 -  Strong Influence
4

Enter Rated value for Complex Processing:
0 - Not Present         1 - Insignificant       2 -  Moderate  3 - Average     4 -  Significant       5 -  Strong Influence
4

Enter Rated value for Re-usability:
0 - Not Present         1 - Insignificant       2 -  Moderate  3 - Average     4 -  Significant       5 -  Strong Influence
2

Enter Rated value for Installation ease:
0 - Not Present         1 - Insignificant       2 -  Moderate  3 - Average     4 -  Significant       5 -  Strong Influence
3

Enter Rated value for Operational ease:
0 - Not Present         1 - Insignificant       2 -  Moderate  3 - Average     4 -  Significant       5 -  Strong Influence
1

Enter Rated value for Multiple sites:
0 - Not Present         1 - Insignificant       2 -  Moderate  3 - Average     4 -  Significant       5 -  Strong Influence
5
```



Findings & Learning: -

- We have successfully implemented class point method and calculated CP1 and CP2.
- We have also calculated the efforts corresponding to each value of CP1 and CP2.
- We learnt strength and weaknesses of class point method.