**Міністерство освіти і науки України**

**Національний університет «Запорізька Політехніка»**

Кафедра програмних засобів

# ЗВІТ

з лабораторної роботи №5

з дисципліни «Моделювання та Аналіз Програмного Забезпечення» на тему:

«Моделювання довільних дисциплін обслуговування з використанням лангюгів користувача; вивчення принципів побудови гістограм»

**Виконав:**

Студент групи КНТ-122 О. А. Онищенко

**Прийняли:**

Викладач: Ж. К. Камінська

2024

# МОДЕЛЮВАННЯ ДОВІЛЬНИХ ДИСЦИПЛІН ОБСЛУГОВУВАННЯ З ВИКОРИСТАННЯМ ЛАНГЮГІВ КОРИСТУВАЧА; ВИВЧЕННЯ ПРИНЦИПІВ ПОБУДОВИ ГІСТОГРАМ

## Мета роботи

Метою роботи є освоєння процедур будування вибором випадкових величин та графічного виводу інформації в системі імітаційного моделювання SIMC; вивчення принципів моделювання різних дисциплін, які обслуговуються на основі списків користувача.

## Результати виконання

## Код програми

```
#include "../simc/simc.h"
#include <iostream>
using namespace std;

void one() {
  auto Total_Modeling_Time = 8*60*60;
  auto Last_Time = NULL;

  auto Interval_Min=300-20;
  auto Interval_Max=300+20;

  auto First_Delay_Min=100-20;
  auto First_Delay_Max=100+20;

  auto Second_Delay_Min=110-25;
  auto Second_Delay_Max=110+25;

  /*
  first: first, second
  second: first, second
  third: first
  */

  pfacility First_Worker;
  pfacility Second_Worker;
```

```
  signalp Signal;
  phistogram Interval_Graphics;

  initlist(Total_Modeling_Time);
  initcreate(1, 0);

  newfac(First_Worker, "\"First Worker\"");
  newfac(Second_Worker, "\"Second Worker\"");
  newhist(Interval_Graphics, 177,277, 10, true, "\"Intervals\"");

  while (systime < Total_Modeling_Time) {
    plan();
    switch (sysevent) {
      case 1: create(randab(Interval_Min,Interval_Max,v1)); Last_Time =
systime; break;
      case 2: split(1,8); break;
      case 3: seize(First_Worker); break;
      case 4: delayt(randab(First_Delay_Min,First_Delay_Max,v1)); break;
      case 5: outfac(First_Worker); break;
      case 6: if (Second_Worker->status == facility::seized)
accept(Signal); else send(Signal); break;
      case 7: next(13); break;
      case 8: seize(Second_Worker); break;
      case 9: delayt(randab(Second_Delay_Min,Second_Delay_Max,v1));
break;
      case 10: outfac(Second_Worker); break;
      case 11: if (First_Worker->status == facility::seized)
accept(Signal); else send(Signal); break;
      case 12: next(18); break;
      case 13: seize(First_Worker); break;
      case 14: delayt(randab(First_Delay_Min,First_Delay_Max,v1)); break;
      case 15: outfac(First_Worker); break;
      case 16: if (Second_Worker->status == facility::seized)
accept(Signal); else send(Signal); break;
      case 17: next(22); break;
      case 18: seize(Second_Worker); break;
      case 19: delayt(randab(Second_Delay_Min,Second_Delay_Max,v1));
break;
      case 20: outfac(Second_Worker); break;
      case 21: if (First_Worker->status == facility::seized)
accept(Signal); else send(Signal); break;
      case 22: assemble(2); break;
      case 23: tabulate(Interval_Graphics, systime - Last_Time);
destroy(); break;
    }
  }
  printall();
```

```
  cout << "Modeling finished, glory to Jesus Christ our Holy Lord GOD
Almighty King of Kings and Lord of Lords" << endl << endl;
}

void two() {
  auto Patient_Interval_Min=6-1;
  auto Patient_Interval_Max=6+1;

  auto Doctor_Interval_Min=7-2;
  auto Doctor_Interval_Max=7+2;

  auto Total_Modeling_Time = 3*60;
  auto Total_Patients = 0;

  pfacility Doctors_Office;
  plistt Doctors_Queue;

  initlist(Total_Modeling_Time);
  initcreate(1, 0);

  newfac(Doctors_Office, "\"Doctors Office\"");
  newuserlt(Doctors_Queue, "\"Doctor Queue\"");

  while (systime < Total_Modeling_Time) {
    plan();
    switch (sysevent) {
      case 1: create(randab(Patient_Interval_Min, Patient_Interval_Max,
v1)); break;
      case 2: trans->testprty = false; inlfifo(Doctors_Queue);
outuserlt(Doctors_Queue); break;
      case 3: seize(Doctors_Office); break;
      case 4: delayt(randab(Doctor_Interval_Min, Doctor_Interval_Max,
v1)); break;
      case 5: outfac(Doctors_Office); break;
      case 6: Total_Patients+=1; destroy(); break;
    }
  }

  cout << "Total Patients: " << Total_Patients << endl;
  printall();
}

int main()
{
  one();
  two();
  return 0;
}
```

## Виконання програми

| CIM-CI++ v1.2 | НУ"ЗП" | 2024 |
|---|---|---|
| Ж. К. Камінська<br>С. М. Сердюк | | |

Загальні параметри середовища:

| | |
|---|---|
| Поточний час | 28837.899 |
| Поточна подія | 1 |
| Поточний транзакт | 2 |
| Усього подій | 2305.000 |
| Час моделювання | 0.00 сек. |
| Середній час виконання події | 0.00000 сек/подія |

| ПОДІЯ | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|
| УСЬОГО | 97 | 96 | 96 | 96 | 96 | 96 | 96 | 96 | 96 | 96 |

| ПОДІЯ | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 |
|---|---|---|---|---|---|---|---|---|---|---|
| УСЬОГО | 96 | 96 | 96 | 96 | 96 | 96 | 96 | 96 | 96 | 96 |

| ПОДІЯ | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 |
|---|---|---|---|---|---|---|---|---|---|---|
| УСЬОГО | 96 | 192 | 96 | | | | | | | |

Прилади

| Прилад | Кількість входжень | Середній час обробки | Завантаження | Кількість захоплень | Стан |
|---|---|---|---|---|---|
| "First Worker" | 192 | 98.573 | 0.659 | 0 | FREE |
| "Second Worker" | 192 | 111.207 | 0.743 | 0 | FREE |

**Гістограма "Intervals"**

| | |
|---|---|
| Діапазон вимірювань | 177.000 .. 277.000 |
| Загальна кількість входжень | 96 |
| Середнє значення | 227.87732 |
| Дисперсія | 245.87641 |
| Число виходів за інтервал зліва | 0 |
| Число виходів за інтервал справа | 0 |

Таблиця спостережень

| | | | | | |
|---|---|---|---|---|---|
| 177.0000 .. 188.1111 | 1 | 188.1111 .. 199.2222 | 1 | 199.2222 .. 210.3333 | 11 |
| 210.3333 .. 221.4444 | 20 | 221.4444 .. 232.5556 | 26 | 232.5556 .. 243.6667 | 21 |
| 243.6667 .. 254.7778 | 12 | 254.7778 .. 265.8889 | 3 | 265.8889 .. 277.0000 | 1 |

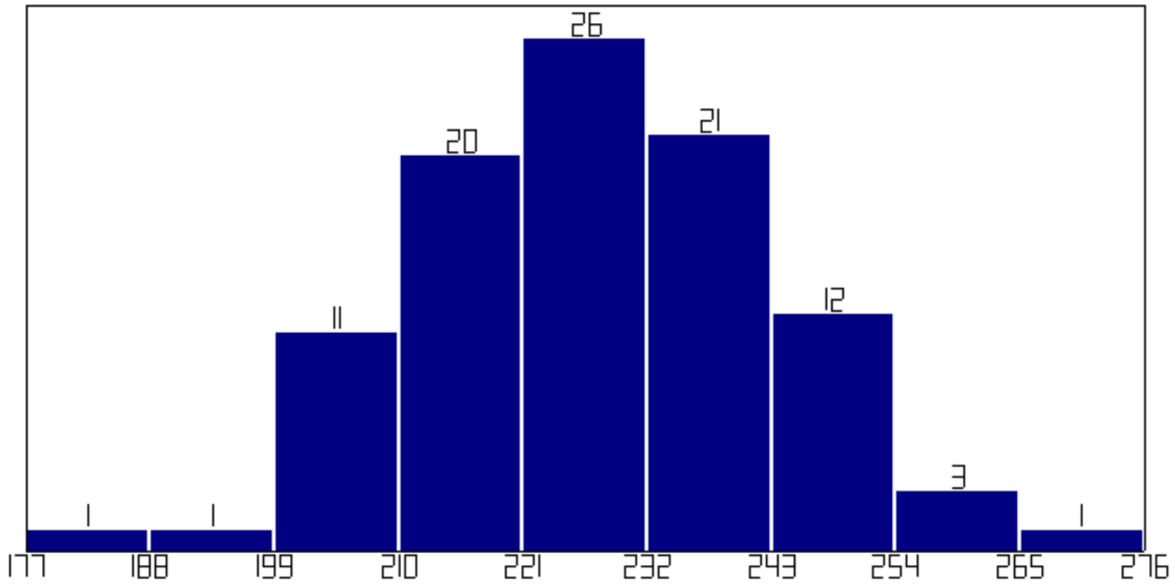


Рисунок 1.1 – Загальне завдання – браузер

```
 Enter a name for the Report HTML file: gen
Modeling finished, glory to Jesus Christ our Holy Lord GOD Almighty King of Kings and Lord of Lords

 Enter a name for the Report HTML file: gen
Total Patients: 25
```

Рисунок 1.2 – Загальне завдання – консолька ✝

## Висновки

Таким чином ми освоїли процедури будування вибором випадкових величин та графічного виводу інформації в системі імітаційного моделювання SIMC, а також вивчили принципи моделювання різних дисциплін, які обслуговуються на основі списків користувача

## Контрольні питання

### Процедури створення гістограм в SIMC

Побудова гістограм у системі імітаційного моделювання SIMC відбувається за допомогою користувацького типу phistogram. У самій моделі посилатися на нього можна черезе phistogram h. Для ініціалізації нової гістограми використовується функція void newhist(phistogram, double, double hint2, bool, alfa); де перший параметр - посилання на гістограму, другий - нижня межа, третій - верхня межа, четвертий - число інтервалів, п'ятий - ключ печатки, шостий - навзва гістограми. Метод табулювання має наступний формат: void tabulate(phistogram hist, double r), де hist - посилання на гістограму, а r - табульоване значення

### Процедури блокування транзактів в SIMC

У системі імітаційного моделювання SIMC існують два способи блокування транзактів: через функцію wait або через методи accept/send. Функція wait очікуватиме поки у системі не відбудеться зазначена подія. Функція має наступний вигляд: void wait(event e), де e - номер події на яку чекаємо. Методи accept/send мають наступний вигляд: void accept(signal s),

void send(signal s). У SIMC визначений тип сигнал signal, який приймає значненя від 1 до signmax.

### Процедури збору статистики в SIMC

У системі імітаційного моделювання SIMC імплементовано автоматичний збір статистики під час виконання програми, однак існують способи виводу та видалення статистики вручну через відповідні методи. Такі методи включають: void resetall() для скидання всієї статистики, що була накопичена під час моделювання, а також void clear() аби повернути усі транзакти у delist та скинути всю поточну статистику.