Zadanie Programistyczne nr 2

15.01.2020r.

Imię: Jakub

Nazwisko: Kowalczyk

Prowadzący: dr Inż. Marek Niewiński

Treść zadania nr 6: Napisać program symulujący zachowanie się klientów w oddziale banku. Przyjąć następujące założenia:

Klienci przychodzą do banku losowo (według zadanego rozkładu prawdopodobieństwa – np. rozkładu normalnego opisanego średnią i odchyleniem standardowym) i ustawiają się w najkrótszej kolejce:

Czas obsługi klienta przez urzędniczkę w każdym okienku zadany przez kolejne zadane rozkłady prawdopodobieństwa.

Program powinien wyznaczać:

1. Liczbę osób w kolejkach,
2. Liczbę obsłużonych klientów przez poszczególne okienka,
3. Średni czas przebywania obsłużonych klientów w systemie.

Wielkości te powinny być wyznaczane dla wybranych przedziałów czasowych

Uwaga: program wczytuje wszystkie dane sterujące z plików.

Dane należy przechowywać z użyciem dynamicznej alokacji pamięci.

Działanie programu: Program symuluje kolejki do kas w banku. Dane sterujące, są wczytywane z pliku „WindowsTimeAverageSigmaCheck”, takie jak :

1. Ilość kas.
2. Czas symulacji.
3. Średnia wartość – (rozkład normalny gaussa).
4. Sigma– (rozkład normalny gaussa).
5. Czas sprawdzania stanu kas.

Dane te powinny być wpisywane kolejno po wciśnięciu „Enter”.

Program wykorzystuje bibliotekę GSL (gsl/gsl_randist.h)(gsl/gsl_rng.h) w celu obliczenia rozkładu normalnego.

Program kompiluje za pomocą komendy „gcc -Wall prog2.c -lgsl -lgslcblas -lm” , a uruchamiam „./a.out”.

1.Struktury:

Struktury	**Do czego służy**
Client	Złożony typ zmiennych specyficznych dla klientów banku.
Queue	Złożony typ zmiennych dla kolejek
Window	Złożony typ zmiennych dla okienek

2.Funkcje:

Nazwa	Typ zwracany	Co robi
main()	void	Funkcja główna wywołująca kolejne funkcje.
printClients(struct , int)	void	Wypisuje klientów w kolejkach do każdego okna.
timeOfService(struct, gsl_rng *r)	void	Przydziela klientom ich czas obsługi przy okienku.
createAndDistributeClients(int,struct,int,int,gsl_rng *r)	void	Funkcja sprawdza, która kolejka jest najkrótsza i wprowadza do niej nowego klienta.
deQueueClients(struct,int)	void	Wypuszcza obsłużonego klienta z systemu.
incrementTimeOfService(struct,int,int)	void	Funkcja inkrementuje zmienne czasowe u każdego klienta oraz sprawdza czy któryś z klientów. Nie powinien być wyjęty z systemu.
createWindows(int,struct)	void	Tworzy zadaną liczbę okienek i inicjuje jej wartości.
destruktor(int,struct)	void	Zwalnia zaalokowaną pamięć.

gsl_rng_alloc(gsl_rng_taus)	int	Zwraca wskaźnik do nowo stworzonego generatora losowych liczb pewnego typu
gsl_rng_gaussian(int,int)	float	Zwraca zmienną losową Gaussa, ze średnią i sigmą wyznaczoną przez użytkownika

3.Zmienne:

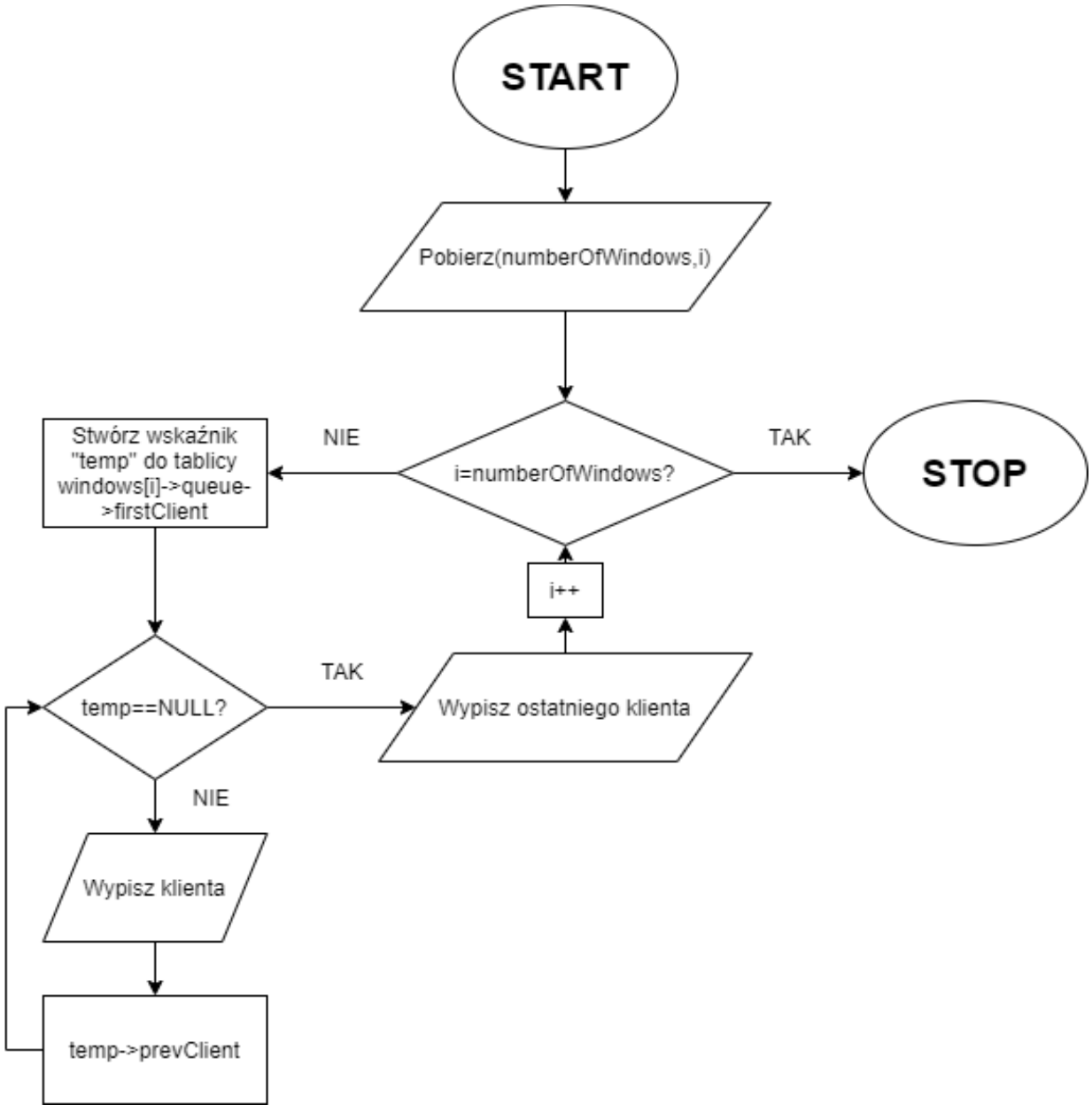
Nazwa	Typ	Funkcja/Struktura	Do czego służy
timeInQueue	int	printClients, createAndDistributeClients, deQueueClients, incrementTimeOfService, Client	Przypisywana jest mu zmienna czasowa wyznaczająca czas stania klienta w kolejce
currentServiceTime	int	printClients, timeOfService, createAndDistributeClients, incrementTimeOfService, Client	Przechowuje zmienną czasową informującą o danym czasie obługi klienta przy okienku.
serviceTime	int	printClients, timeOfService, createAndDistributeClients, deQueueClients, incrementTimeOfService, Client	Przechowuje zmienną wyznaczającą długość obsługi klienta przy okienku.
ID	int	printClients, createWindows, Window	Przechowuje zmienną z numerem identyfikacyjnym okienka.
peopleServed	int	printClients, incrementTimeOfService, createWindows, Window	Przechowuje zmienną z informacją o ilości obsłużonych klientów dla poszczególnych okienek.

numberOfClients	unsigned int	printClients, createAndDistributeClients, deQueueClients, createWindows, Queue	Przechowuje zmienną z informacją o ilości klientów w poszczególnej kolejce.
numberOfWindows	int	printClients, createAndDistributeClients, incrementTimeOfService, createWindows, destruktor, main,	Przechowuje zmienną o Ilości okienek w symulacji.
i	int	createAndDistributeClients, incrementTimeOfService, createWindows, destruktor	Zmienna do inkrementacji w pętlach.
j	int	printClients, createAndDistributeClients, createWindows,	Zmienna do inkrementacji w pętlach.
time	int	timeOfService	Zmienna do przechowania czasu obsługi klienta.
min	int	createAndDistributeClients,	Zmienna użyta do zapamiętania najkrótszej kolejki.
simulationTime	int	main	Przechowuje czas symulacji wzięty z pliku.
sigma	int	main	Przechowuje wartość sigmy wziętej z pliku.
average	int	main	Przechowuje wartość średniej wziętej z pliku.
currentSimulationTime	int	main	Przechowuje zmienną wartości aktualnego czasu symulacji.

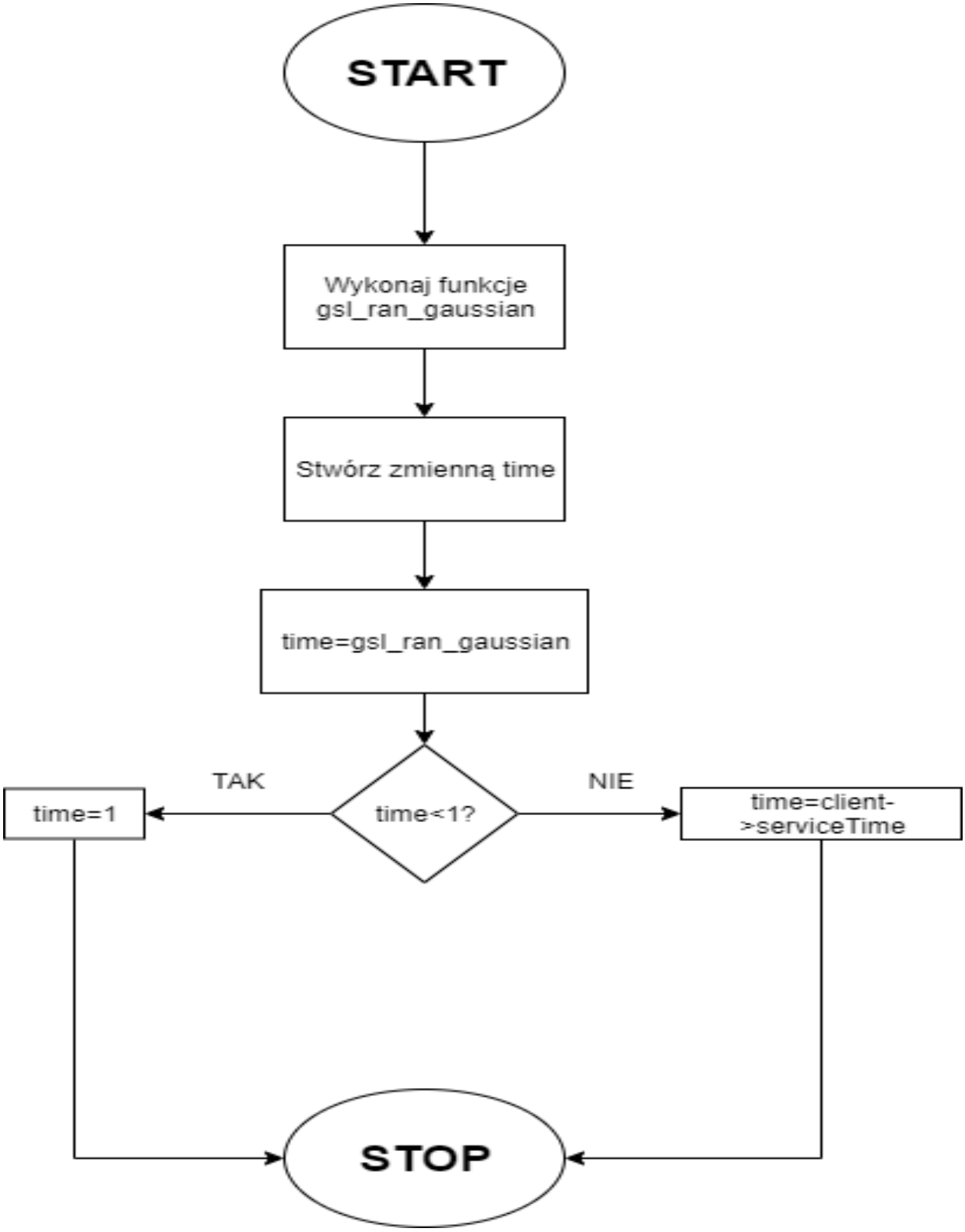
checkState	int	main	Przechowuje zmienną z wartością co ile program ma sprawdzić stan kolejek.
clientsEntered	double	main	Przechowuje zmienną z ilością wchodzących nowych klientów.

4.Schematy Blokowe:

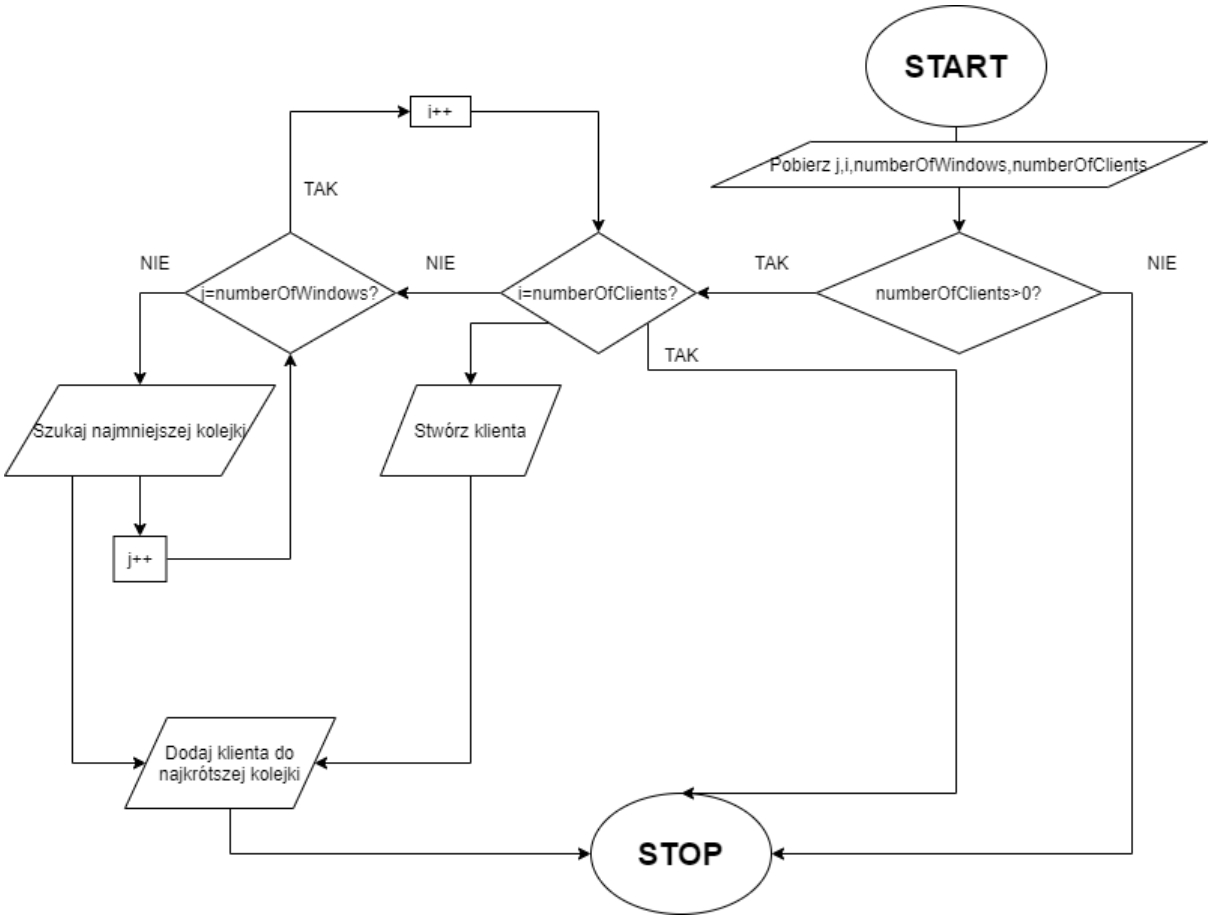
printClients:



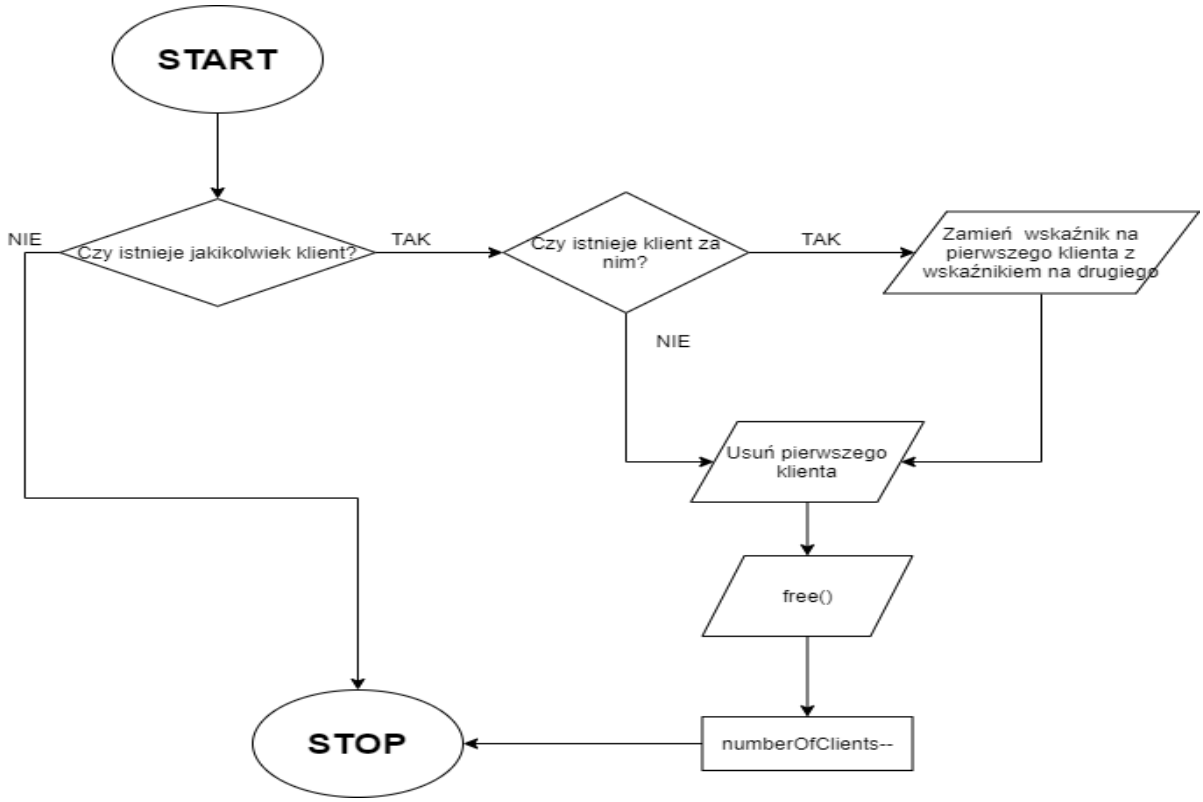
timeOfService:



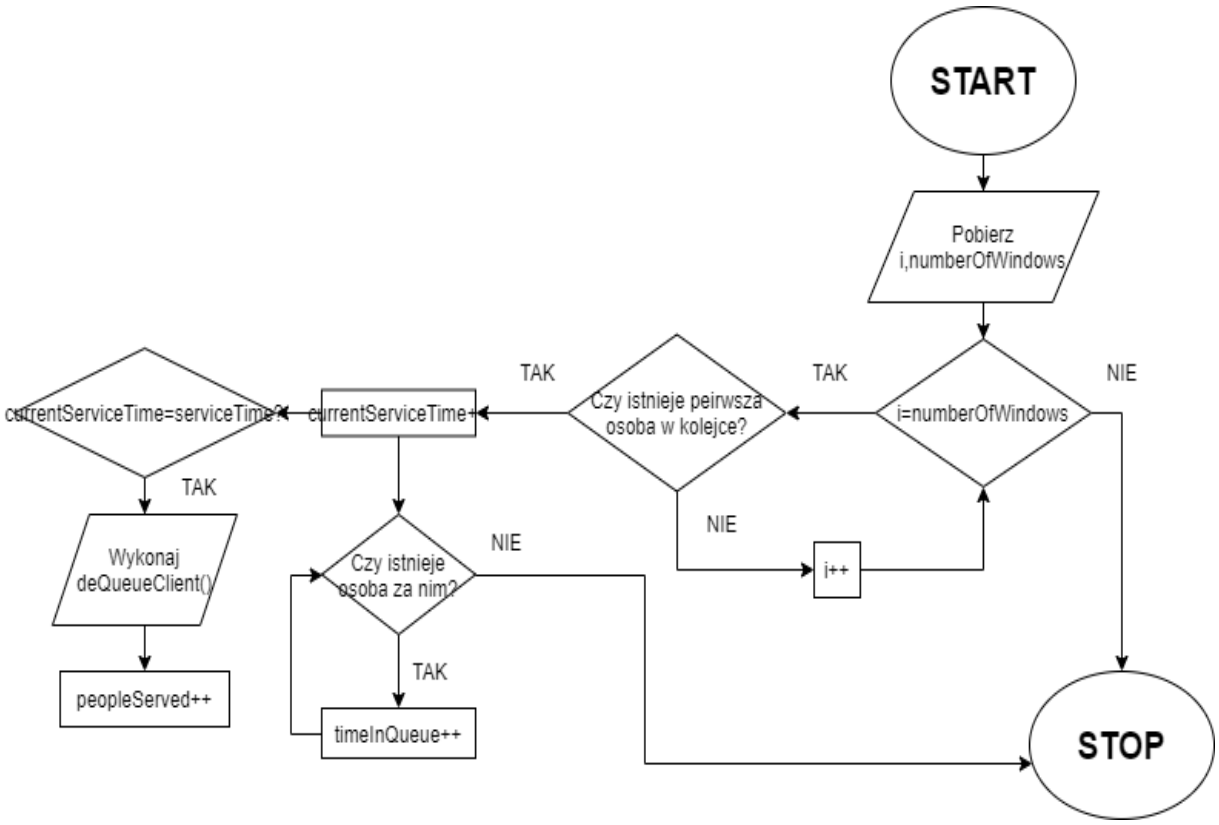
createAndDistribute:



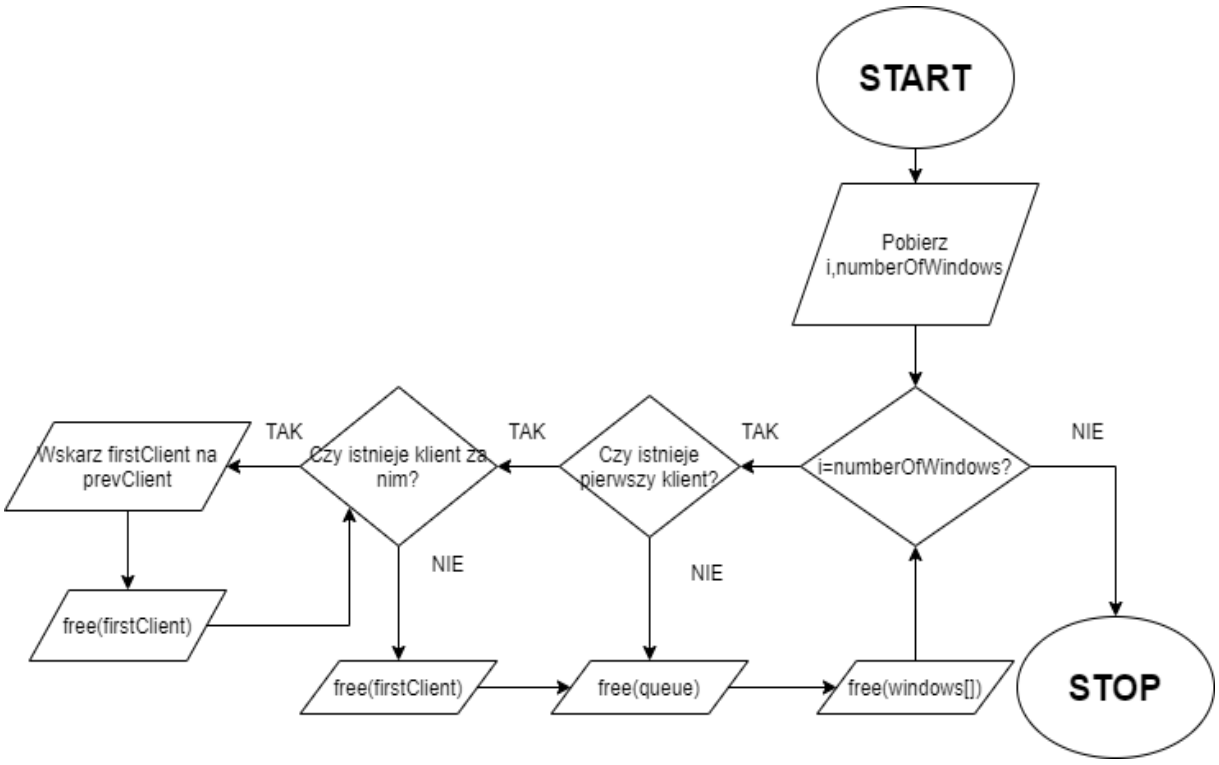
deQueueClients:



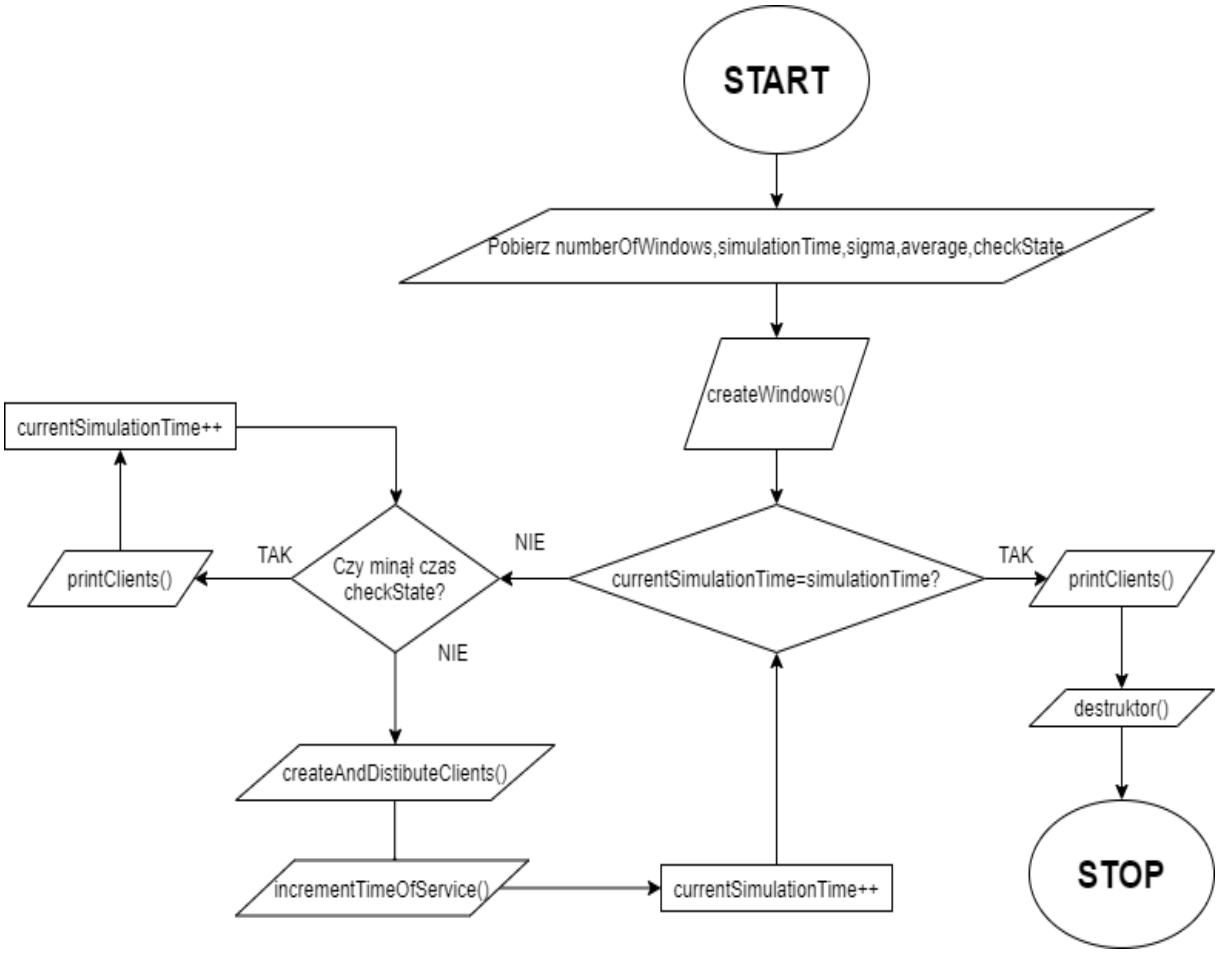
incrementTimeOfService:



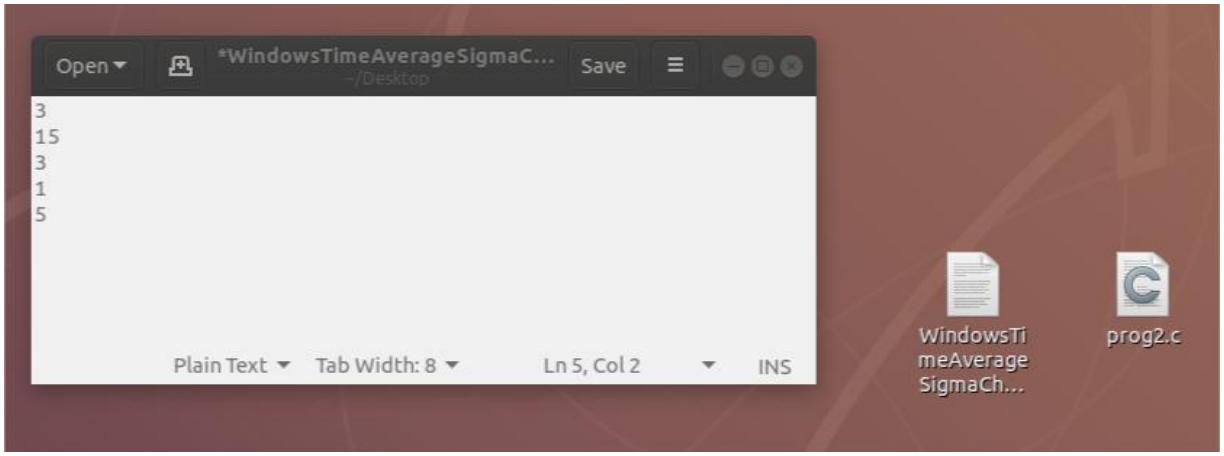
destruktor:



main:



5.Przykładowe działanie programu



```
jakub@jakub-VirtualBox:~/Desktop$ gcc -Wall prog2.c -lgsl -lgslcblas -lm
jakub@jakub-VirtualBox:~/Desktop$ ./a.out
Number of windows: 3 , Simulation Time: 15 , Average: 3 , Sigma: 1, Check the state of windows every: 5
--------------------
Window 1 Clients Left: 0 Clients served: 0 AVG -nan:
--------------------
Window 2 Clients Left: 0 Clients served: 0 AVG -nan:
--------------------
Window 3 Clients Left: 0 Clients served: 0 AVG -nan:
--------------------
```

```
--------------------
Window 1 Clients Left: 9 Clients served: 3 AVG 7.000:
Client 0x56426700ba40: Service Time: 4 Current Service Time:0 Time in Queue: 10
Client 0x56426700bbf0: Service Time: 5 Current Service Time:0 Time in Queue: 11
Client 0x56426700ba70: Service Time: 3 Current Service Time:0 Time in Queue: 9
Client 0x56426700bd10: Service Time: 2 Current Service Time:0 Time in Queue: 9
Client 0x56426700bda0: Service Time: 6 Current Service Time:0 Time in Queue: 8
Client 0x56426700bb00: Service Time: 1 Current Service Time:0 Time in Queue: 7
Client 0x56426700be00: Service Time: 5 Current Service Time:0 Time in Queue: 6
Client 0x56426700bec0: Service Time: 8 Current Service Time:0 Time in Queue: 4
Client 0x56426700bf80: Service Time: 8 Current Service Time:0 Time in Queue: 1
--------------------
Window 2 Clients Left: 10 Clients served: 3 AVG 6.333:
Client 0x56426700ba10: Service Time: 3 Current Service Time:2 Time in Queue: 8
Client 0x56426700bbc0: Service Time: 4 Current Service Time:0 Time in Queue: 11
Client 0x56426700bc80: Service Time: 4 Current Service Time:0 Time in Queue: 9
Client 0x56426700bd40: Service Time: 4 Current Service Time:0 Time in Queue: 8
Client 0x56426700bd70: Service Time: 8 Current Service Time:0 Time in Queue: 8
Client 0x56426700bad0: Service Time: 2 Current Service Time:0 Time in Queue: 7
Client 0x56426700be30: Service Time: 4 Current Service Time:0 Time in Queue: 6
Client 0x56426700bef0: Service Time: 1 Current Service Time:0 Time in Queue: 4
Client 0x56426700bb60: Service Time: 5 Current Service Time:0 Time in Queue: 2
Client 0x56426700bf20: Service Time: 2 Current Service Time:0 Time in Queue: 2
--------------------
Window 3 Clients Left: 10 Clients served: 3 AVG 8.333:
Client 0x56426700bc20: Service Time: 4 Current Service Time:1 Time in Queue: 8
Client 0x56426700bc50: Service Time: 5 Current Service Time:0 Time in Queue: 10
Client 0x56426700bcb0: Service Time: 4 Current Service Time:0 Time in Queue: 9
Client 0x56426700bce0: Service Time: 5 Current Service Time:0 Time in Queue: 9
Client 0x56426700bdd0: Service Time: 5 Current Service Time:0 Time in Queue: 7
Client 0x56426700be60: Service Time: 1 Current Service Time:0 Time in Queue: 5
Client 0x56426700be90: Service Time: 5 Current Service Time:0 Time in Queue: 5
Client 0x56426700baa0: Service Time: 5 Current Service Time:0 Time in Queue: 3
Client 0x56426700bb90: Service Time: 3 Current Service Time:0 Time in Queue: 1
Client 0x56426700bf50: Service Time: 5 Current Service Time:0 Time in Queue: 1
--------------------
jakub@jakub-VirtualBox:~/Desktop$
```

6.Wnioski

Program spełnia wszystkie wymogi zadania. Działa nawet przy bardzo wysokich wprowadzonych zmiennych. Choć program nie zawiesza się to preferowane jest użycie małej wartości Average i Sigma gdyż kolejki wypełniają się bardzo szybko przez co trudno zrozumieć co program wykonał pomiędzy sprawdzeniem stanu kolejek. Innym problemem jest metoda wprowadzania zmiennych do odczytania z pliku. W moim programie jest wymagane by osoba wprowadzająca dane do pliku wpisała je poprawnie (w odpowiedniej kolejności i po Enterze), co może byś niezrozumiałe dla niektórych użytkowników.