ASD - ćwiczenia I

Zasady zaliczenia przedmiotu

- nieusprawiedliwione nieobecności – 2 szt.,
- wejściówki co dwa tygodnie – 7 szt. × 5 pkt.,
- kolokwium I w połowie listopada (prawdopodobnie sobota) – 20 pkt.,
- kolokwium II na ostatnich ćwiczeniach – 30 pkt.,
- aktywność na ćwiczeniach, prace domowe itp. – 15 pkt.
- razem 100 pkt., ze skalą ocen:

punkty	ocena
91 - 100	5
81 - 90	4,5
71 - 80	4
61 - 70	3,5
51 - 60	3
0 - 49	2

- obowiązkowe zadanie domowe co dwa tygodnie,
- dla chętnych „duży“ projekt za dodatkowe punkty.
- terminarz zajęć (Z – zadanie domowe, W – wejściówka, K – kolokwium):

zajęcia	co będzie?	zajęcia	co będzie?
I	**Z**	IX	**Z**
II	**W**	X	**W**
III	**Z**	XI	**Z**
IV	**W**	XII	**W**
V	**Z**	XIII	**Z**
VI	**W**	XIV	**W**
VII	**Z**	XV	**K**
VIII	**W**		

Zadanie o trójkątach

Niech S będzie zbiorem $n > 2$ odcinków parami różnej i niezerowej długości $S[1], S[2], \ldots, S[n]$, rozłożonych na płaszczyźnie euklidesowej, którego elementy opisane są za pomocą poniższej struktury danych:

```
typedef str_Segment Segment;

struct str_Segment {
  real x1, x2, y1, y2;
};
```

Napisz możliwie efektywną funkcję

```
bool TEST(Segment S[], int n),
```

która sprawdzi, czy przesuwanie i obracanie dowolnych trzech odcinków $S[p], S[q], S[r] \in S$, może prowadzić do utworzenia trójkąta o niezerowej powierzchni.

Rzędy wielkości funkcji

Zakładamy domyślnie, że:

- dziedzina rozważanych funkcji – zbiór $\mathbb{N}$,
- przeciwdziedzina rozważanych funkcji – zbiór $\mathbb{R}^+$.

Zadania:

1. Uporządkuj malejąco następujące funkcje zmiennej n względem ich rzędów:
   (a) $f_1(n) = \log(n^3)$, $f_2(n) = n\log(n!)$, $f_3(n) = n^{1/3} + n$,
   (b) $f_1(n) = n^3\log(3n)$, $f_2(n) = 2^{\log(n^4)}$, $f_3(n) = \left(\sqrt{2}\right)^n$,
   (c) $f_1(n) = n^{\frac{1}{2}} + \log(n)$, $f_2(n) = \log(n^2)$, $f_3(n) = 2^{\log(n)}$, $f_4(n) = n^2\left(n + n^{\frac{2}{3}}\right)^{\frac{1}{2}}$,
   (d) $f_1(n) = n(\sin(n))^2$, $f_2(n) = n^3\log^3(n)$, $f_3(n) = n^n$, $f_4(n) = 2^{2n}$,
   (e) $f_1(n) = n!$, $f_2(n) = \log^*(n^2)$, $f_3(n) = n^{\frac{3}{2}n}$, $f_4(n) = \log^{\frac{1}{4}}(n^{16})$,
   (f) $f_1(n) = n$, $f_2(n) = \log(n!)$, $f_3(n) = 2^{(n)}$, $f_4(n) = 2^{2n}$.

Zadanie o kieszonkowcu i przekupnych policjantach (do domu)

W kolejce po bilety do teatru stoi $n \geq 1$ osób $C[1], C[2], \ldots, C[n]$, spośród których p to zwykli klienci a q to nieumundurowani policjanci. Osoby stojące w kolejce reperezentowane są za pomocą struktury danych `Client` następującej postaci:

```
typedef str_Client Client;

struct str_Client {
  enum type = {civilian, policeman};
  int m;
};
```

której argumenty to kolejno:

- *type* = *civilian* – flaga określająca fakt, że dany klient jest w rzeczywistości cywilem,
- *type* = *policeman* – flaga określająca fakt, że dany klient jest w rzeczywistości nieumundurowanym policjantem,
- jeżeli *type* = *civilian*, to m – ilość pieniędzy w portfelu danego klienta-cywila,
- jeżeli *type* = *policeman*, to m – wysokość „łapówki" jaką należy dać danemu klientowi-policjantowi aby przymknął on oko na próbę popełnienia przestępstwa o tzw. niskiej szkodliwości społecznej, np. kradzież portfela.

Od dłuższego czasu kolejkę klientów teatru obserwuje kieszonkowiec o pseudonimie „Pusty Portfel", który zamierza okraść portfele kilku osób spośród zbioru $C[1], C[2], \ldots, C[n]$. Zakładamy, że:

- „Pusty Portfel“ dysponuje wiedzą na temat zasobności portfeli klientów teatru jak i na temat wysokości łapówek jakie należy przekazać poszczególnym policjantom w przypadku wykrycia próby kradzieży,
- każdy policjant może przyłapać kieszonkowca tylko wtedy, kiedy próbuje on ukraść jego portfel,
- przekazanie łapówek poszczególnym policjantom odbywa się zawsze następnego dnia po przyłapaniu kieszonkowca,
- „Pusty Portfel“ zawsze kradnie według metody „kilku kolejnych“, która polega na tym, że z kolejki klientów wybiera dwóch $C[i], C[j]$, gdzie $i \leq j$, a następnie próbuje okraść kolejno wszystkie osoby od $C[i], C[i+1], \ldots, C[j-1], C[j]$ włącznie,

Zaprojektuj możliwie efektywną procedurę

`MAX(Client C[], int n),`

która pomoże kieszonkowcowi w określeniu dwóch skrajnych klientów $C[i], C[j]$ oraz wyznaczy ostateczny zarobek (zysk z portfeli klientów pomniejszony o wydatki na łapówki dla policjantów) w przypadku wykonania najlepszego z możliwych skoków według metody „kilku kolejnych“.