Analýza čítania zdrojového kódu

Adam Talian

Mgr. Jozef Tvarožek, PhD.

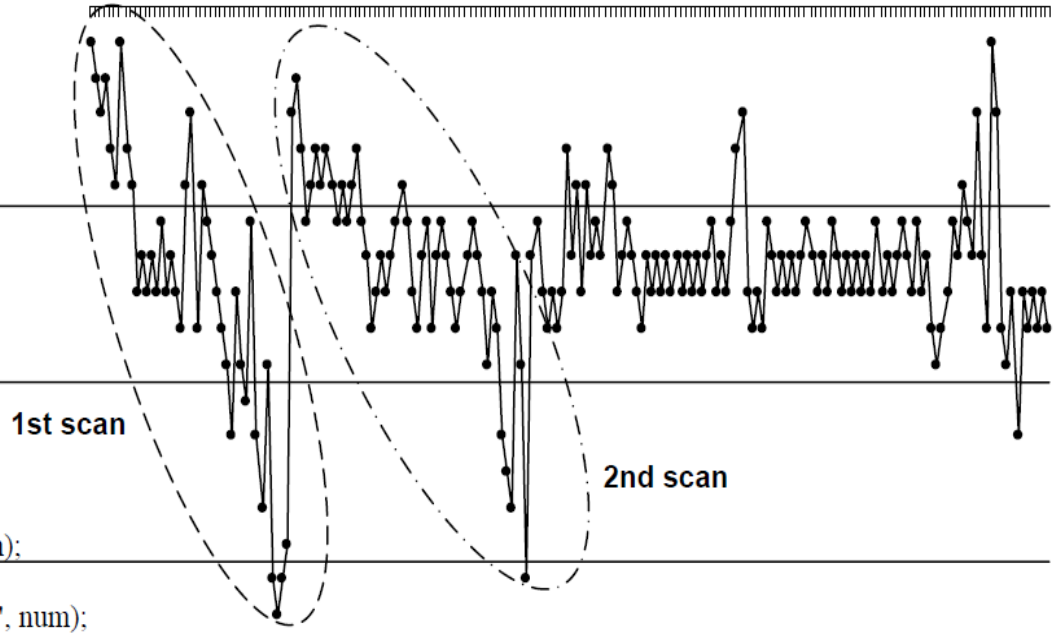
Čo je spravené v oblasti?

- Spôsoby vizualizácie
- Kódovanie – riadok, blok, podblok, signatúry, zoznamy parametrov, volania metód
- Identifikované vzory a stratégie
- Pokusy analýz pohľadov expertov
- Pokusy analýz pohľadov nováčikov

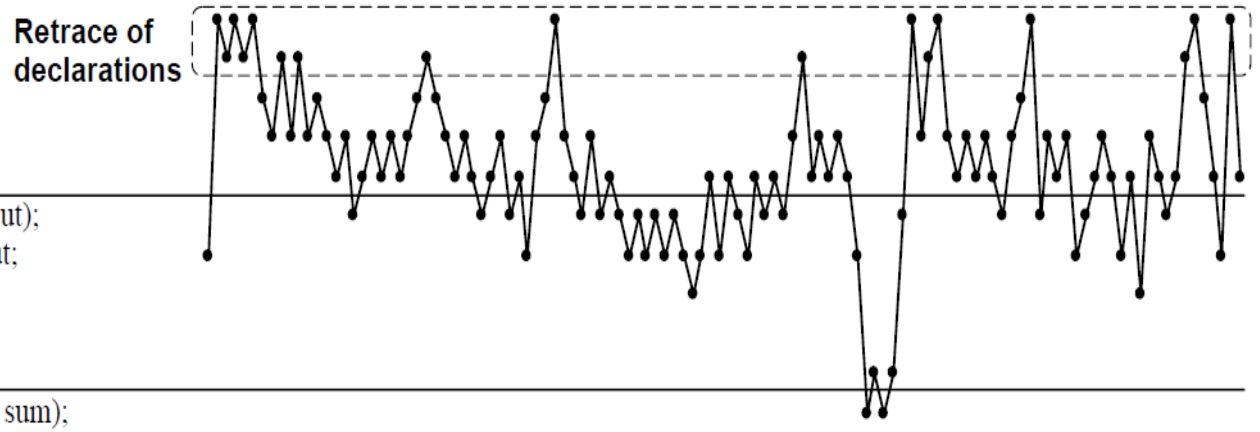
Vzory

- JumpControl
- JustPassingThrough
- Linear
- Scan
- Flicking
- Thrashing
- Signatures

```
01 void main(void){
02    int i, num, isPrime = 0;
03
04    printf("Input Number:");
05    scanf("%d", &num);
06
07    i = 2;
08    while(i < num){
09       if(num%i == 0)
10          isPrime = 1;
11       i = i + 1;
12    }
13
14    if(isPrime == 1)
15       printf("%d is prime number.¥n", num);
16    else
17       printf("%d is NOT prime number.¥n", num);
```




```
01 void main(void){
02    int i, input, sum;
03
04    i = 0;
05    while(i < 5){
06       scanf("%d", &input);
07       sum = sum + input;
08       i = i + 1;
09    }
10
11    printf("Sum:%d¥n", sum);
12 }
```



Stratégie

- AttentionToDetail
- DataFlow
- DesignAtOnce
- TestHypothesis
- Trail&Error
- FlowCycle
- Touchstone
- Wandering
- .. a iné

Ciele

- Vykonať experiment na iných typoch kódov (C a viac logiky)
- Skúsiť aj tam nájsť vzory Scan, JumpControl, Flicking, Thrashing
- Identifikácia problémových miest ale pozerať sa na dáta v kontexte, doteraz viac menej kvantitatívne štatistiky
- Skúsiť sa pozrieť na problematiku z opačnej strany – retrospektívy
- Ideálne navrhnúť metódu, ktorá dokáže automaticky zanalyzovať dáta z eye-trackera a identifikovať problematické miesta v kóde(statická a dynamická analýza kódu).

Čo by sme chceli?

- Pripraviť zdrojové kódy (už nejaké sú) tak aby reflektovali ciele
- Spraviť malý experiment
  - Retrospektívy
  - Identifikácia AOI
- Trocha väčší experiment
  - Bez retrospektív
  - Ideálne len na overenie a kvantitu dát

Thrashing

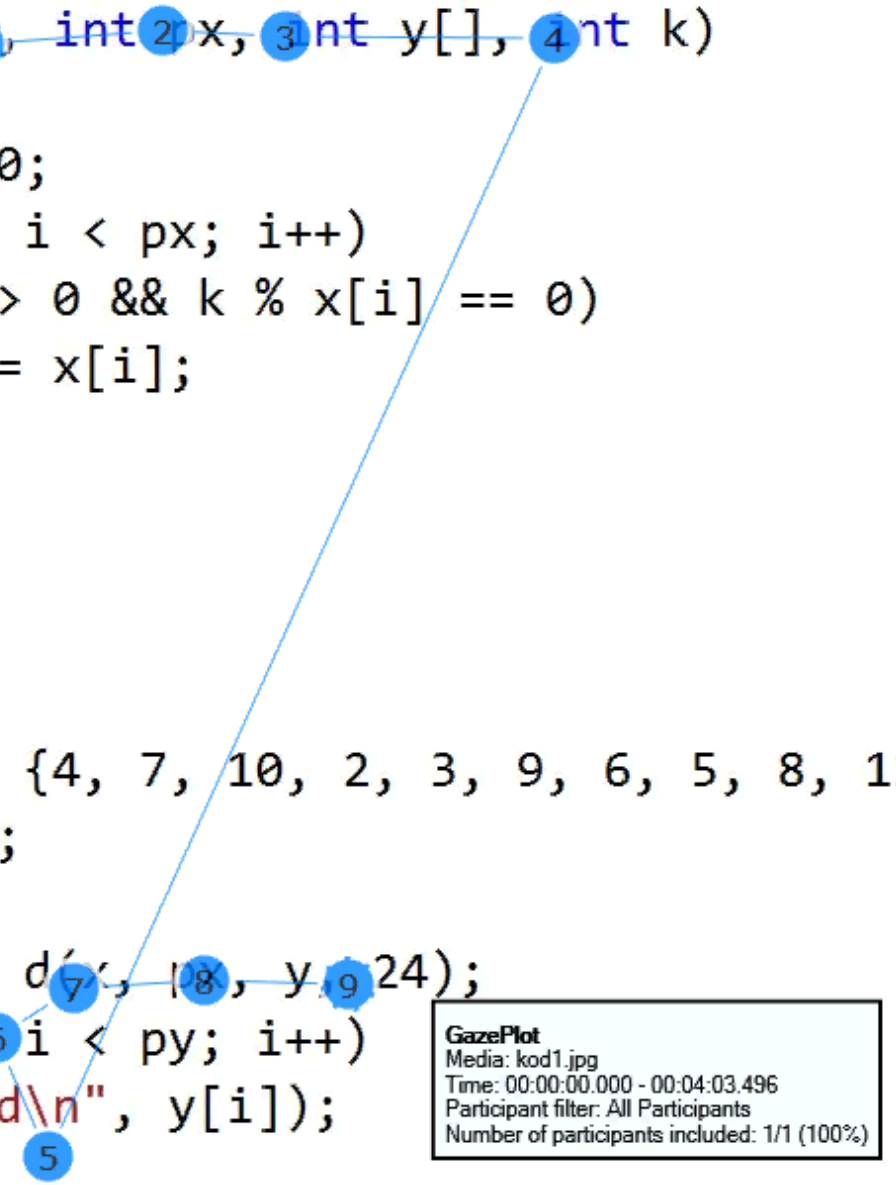
```
13      S *first, *actual, *prev;
14
15      prev = first = (S*)calloc(1, sizeof(S));
16      for(i = 1; i < size + 1; i++) {
17          actual = (S*)calloc(1, sizeof(S));
18          prev = prev->next = actual;
19          actual->value = i;
20      }
```

```
20      }
21
22      actual = first;
23      while(actual->next != NULL) {
24          printf("%d ", actual->value);
25          actual = actual->next;
26      }
```

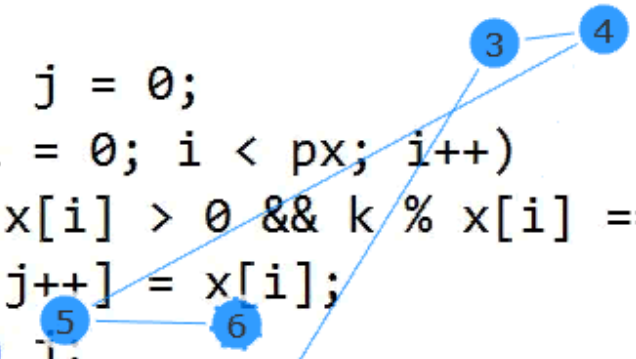
Flicking

- Retrace Declaration
- Retrace Reference

```
#include <stdio.h>

int d(int x[], int px, int y[], int k)
{
  int i, j = 0;
  for (i = 0; i < px; i++)
    if (x[i] > 0 && k % x[i] == 0)
      y[j++] = x[i];
  return j;
}

int main()
{
  int x[10] = {4, 7, 10, 2, 3, 9, 6, 5, 8, 12};
  int px = 10;
  int y[10];
  int i, py = d(x, px, y, 24);
  for (i = 0; i < py; i++)
    printf("%d\n", y[i]);
  return 0;
}
```

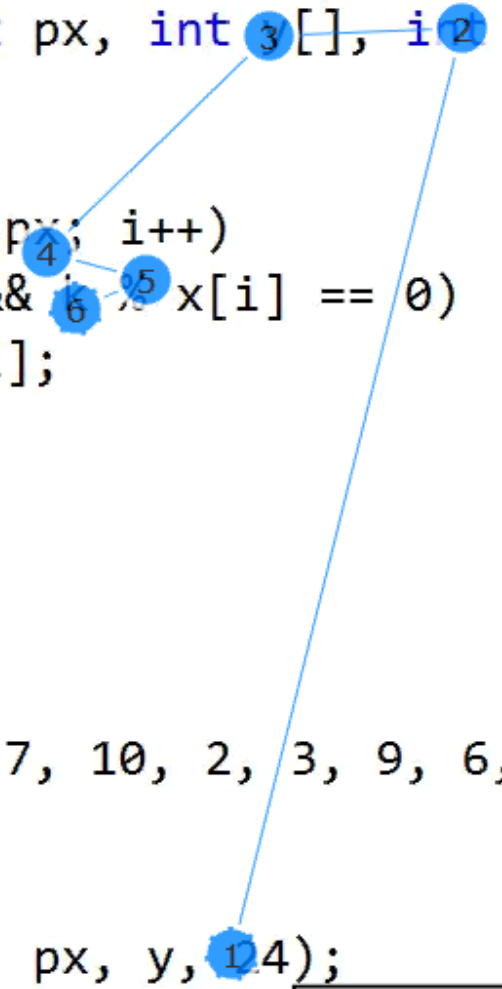


```c
#include <stdio.h>

int d(int x[], int px, int y[], int k)
{
  int i, j = 0;
  for (i = 0; i < px; i++)
    if (x[i] > 0 && k % x[i] == 0)
      y[j++] = x[i];
  return j;
}

int main()
{
  int x[10] = {4, 7, 10, 2, 3, 9, 6, 5, 8, 12};
  int px = 10;
  int y[10];
  int i, py = d(x, px, y, 24);
  for (i = 0; i < py; i++)
    printf("%d\n", y[i]);
  return 0;
}
```



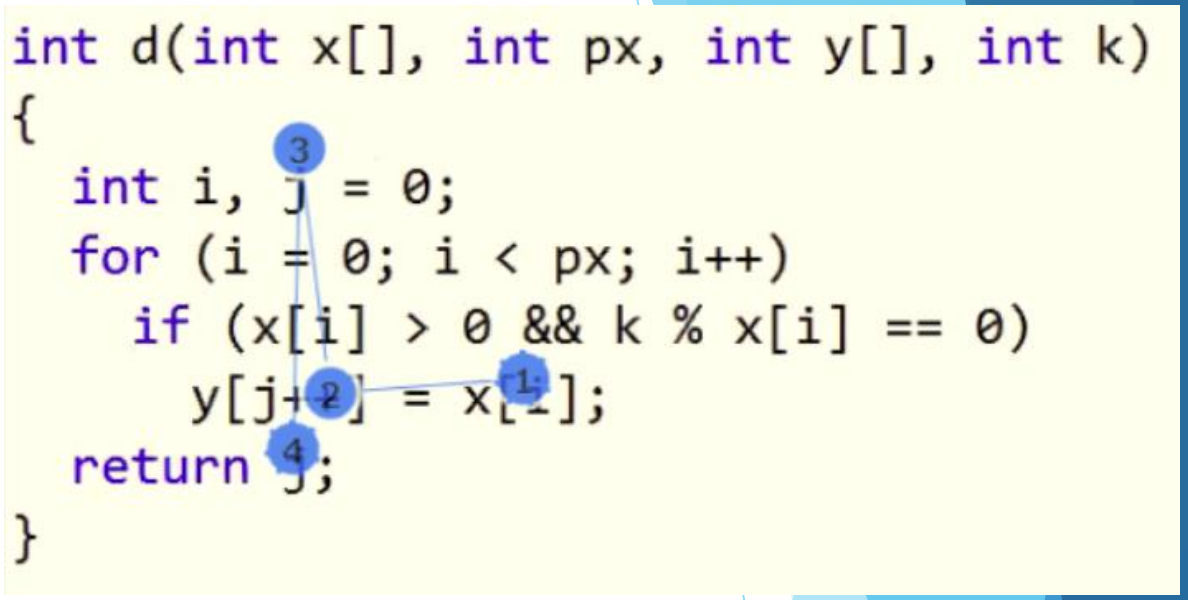
GazePlot
Media: kod1.jpg
Time: 00:00:00.000 - 00:04:03.496
Participant filter: All Participants
Number of participants included: 1/1 (100%)

```c
#include <stdio.h>

int d(int x[], int px, int y[], int k)
{
  int i, j = 0;
  for (i = 0; i < px; i++)
    if (x[i] > 0 && x[i] == 0)
      y[j++] = x[i];
  return j;
}

int main()
{
  int x[10] = {4, 7, 10, 2, 3, 9, 6, 5, 8, 12};
  int px = 10;
  int y[10];
  int i, py = d(x, px, y, 4);
  for (i = 0; i < py; i++)
    printf("%d\n", y[i]);
  return 0;
}
```



GazePlot
Media: kod1.jpg
Time: 00:00:00.000 - 00:04:03.496
Participant filter: All Participants
Number of participants included: 1/1 (100%)

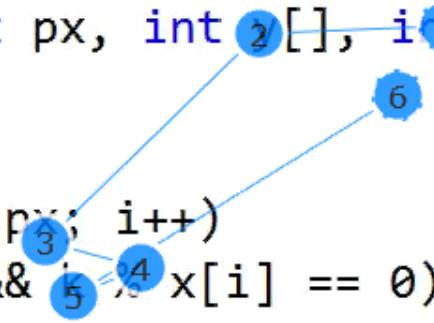
```c
#include <stdio.h>

int d(int x[], int px, int y[], int k)
{
  int i, j = 0;
  for (i = 0; i < px; i++)
    if (x[i] > 0 && k % x[i] == 0)
      y[j++] = x[i];
  return j;
}

int main()
{
  int x[10] = {4, 7, 10, 2, 3, 9, 6, 5, 8, 12};
  int px = 10;
  int y[10];
  int i, py = d(x, px, y, 24);
  for (i = 0; i < py; i++)
    printf("%d\n", y[i]);
  return 0;
}
```



GazePlot
Media: kod1.jpg
Time: 00:00:00.000 - 00:04:03.496
Participant filter: All Participants
Number of participants included: 1/1 (100%)

Náznak JumpControl?

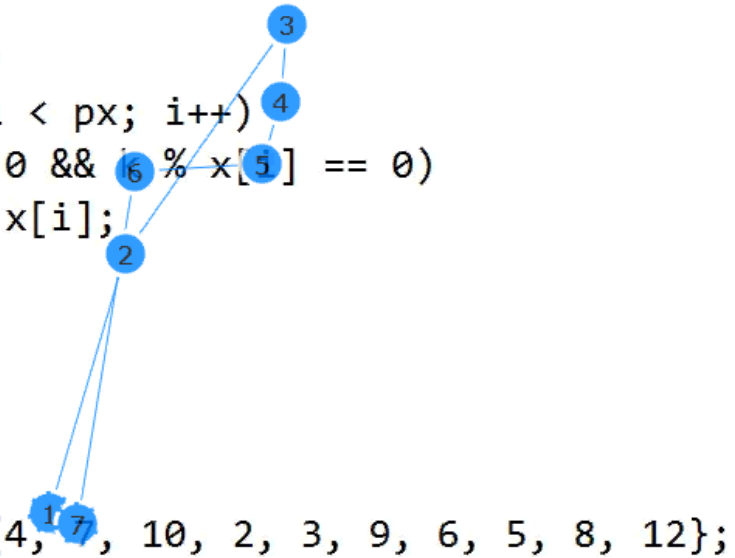
```c
#include <stdio.h>

int d(int x[], int px, int y[], int k)
{
  int i, j = 0;
  for (i = 0; i < px; i++)
    if (x[i] > 0 && x[i] == 0)
      y[j++] = x[i];
  return j;
}

int main()
{
  int x[10] = {4, 7, 10, 2, 3, 9, 6, 5, 8, 12};
  int px = 10;
  int y[10];
  int i, py = d(x, px, y, 24);
  for (i = 0; i < py; i++)
    printf("%d\n", y[i]);
  return 0;
}
```



GazePlot
Media: kod1.jpg
Time: 00:00:00.000 - 00:04:03.496
Participant filter: All Participants
Number of participants included: 1/1 (100%)

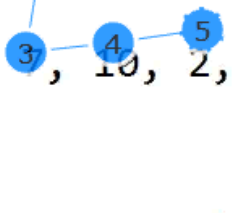
Konečne pochopenie.. ?

```c
#include <stdio.h>

int d(int x[], int px, int y[], int k)
{
  int i, j = 0;
  for (i = 0; i < px; i++)
    if (x[i] > 0 && j % x[i] == 0)
      y[j++] = x[i];
  return j;
}

int main()
{
  int x[10] = {4, 7, 10, 2, 3, 9, 6, 5, 8, 12};
  int px = 10;
  int y[10];
  int i, py = d(x, px, y, 24);
  for (i = 0; i < py; i++)
    printf("%d\n", y[i]);
  return 0;
}
```

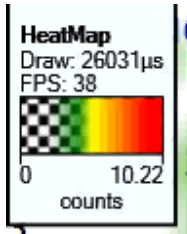


GazePlot
Media: kod1.jpg
Time: 00:00:00.000 - 00:04:03.496
Participant filter: All Participants
Number of participants included: 1/1 (100%)

```c
#include <stdio.h>

int d(int x[], int px, int y[], int k)
{
  int i, j = 0;
  for (i = 0; i < px; i++)
    if (x[i] > 0 && j % x[i] == 0)
      y[j++] = x[i];
  return j;
}

int main()
{
  int x[10] = {4, 7, 10, 2, 3, 9, 6, 5, 8, 12};
  int px = 10;
  int y[10];
  int i, py = d(x, px, y, 24);
  for (i = 0; i < py; i++)
    printf("%d\n", y[i]);
  return 0;
}
```



GazePlot
Media: kod1.jpg
Time: 00:00:00.000 - 00:04:03.496
Participant filter: All Participants
Number of participants included: 1/1 (100%)




```
de <stdio.h>

int x[], int px, int y[], int k)
{
  int i, j = 0;
  for (i = 0; i < px; i++)
    if (x[i] > 0 && x[i] == 0)
      y[
  return ;
}

int main()
{
  int x[ ] = {4, 7, 10, 2, 3, 9, 6, 5, 8, 12};
  int p = ,
  int y[ ];
  int i, = y, 24);
  for (i = 0 i i++)
    printf("%d\n", [i]);
  return 0;
}
```

Podnety do diskusie

- Rady pre experiment
- Ako by mohla hľadaná metóda fungovať? Momentálne vzdialený cieľ.
- Povedal som niečo, čo nedáva zmysel, je to zbytočné alebo nerealizovateľné?
- Hocičo vám napadne, nápady, postrehy, kritika...