# Vzporedni in porazdeljeni sistemi in algoritmi: Vaje

## Uvod v C

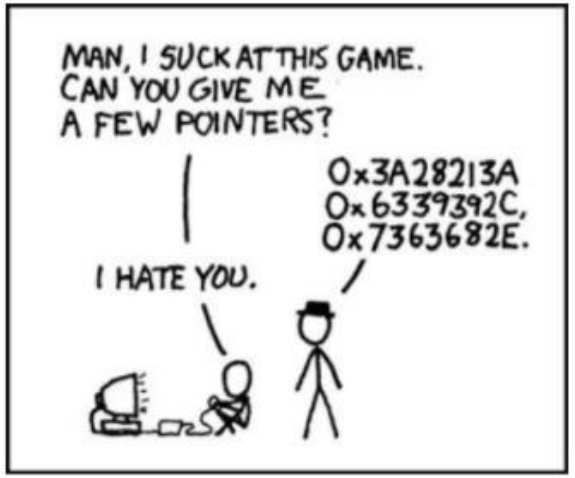
PREDAVATELJ: PATRICIO BULIĆ

ASISTENT: DAVOR SLUGA

# Programski jezik C - ponovitev

Kazalci (pointers) in polja (arrays)

Dinamično zasedanje pomnilnika (dynamic memory allocation)



# Kazalci

Kazalec je spremenljivka, ki hrani naslov pomnilniške lokacije

Operator & (ne zamenjuj z operatorjem AND!)

- Pove naslov pomnilniške lokacije kjer se spremenljivka nahaja

Operator * (ne zamenjuj z množenjem!)

- deklaracija podatkovnega tipa kazalec
- vrne vrednost, ki se nahaja na naslovu na katerega kazalec kaže

# Kazalci - primer

```
int x,y,*p;


x=3;
p=&x;
y=*p;
*p=x+y;
```

Kako se vrednosti x,y in p spreminjajo skozi program?

# Kazalci in polja

```
int seznam[4]={1,2,3,4};
int *p=seznam;
```

Ekvivalentni izrazi:

```
seznam[2]=5;   *(p+2)=5;   p[2]=5;   *(seznam+2)=5;
```

Razlike med polji in kazalci:

- Kazalci hranijo naslov kjer se podatki začnejo, polja pa dejanske podatke
- Kazalci -> posredno naslavljanje
- Polja -> neposredno naslavljanje

# Polja kot argumenti funkcij

- Funkcije v C-ju lahko polje kot argument prevzame le po referenci. Funkcija dobi naslov polja kjer se podatki začnejo.

```
int vsota(int *T, int n){
    int sum = 0;
    while(n--){
        sum += T[n];
    }
    return sum;
}

int main(){
    const int A[] = {0,1,2,3};
    printf("%d\n",vsota(A,(sizeof A)/ sizeof(int)));
}
```

# Dinamično zasedanje pomnilnika

Uporabljamo, če količina potrebnega pomnilnika, ni znana ob prevajanju programa.

Funkcije:

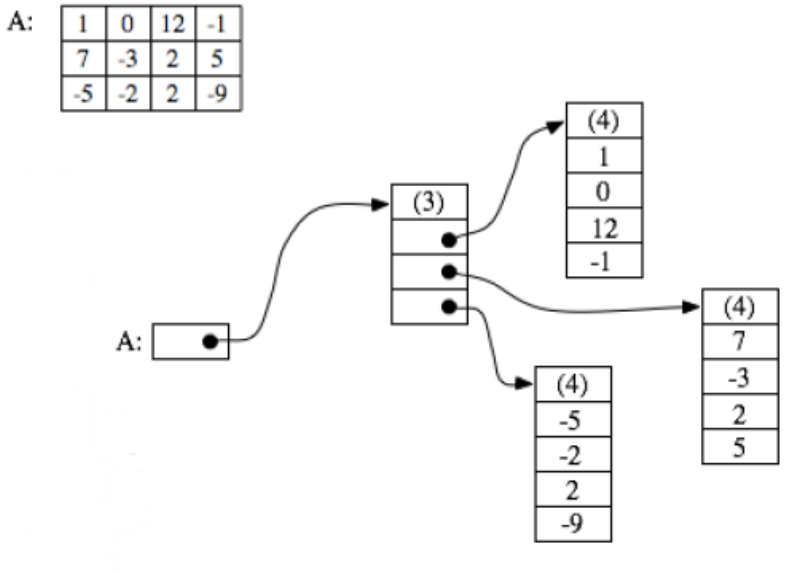
```
void *malloc(size_t size);

void *calloc(size_t nItems, size_t size);

void *realloc(void *ptr, size_t size);

void free(void *ptr);
```

# Dinamično zasedanje pomnilnika

```
int * A;
double * B;
char * C;
int main(){
    A = (int *)malloc(10*sizeof(int));
    B = (double *)calloc(10,sizeof(double));
    C = (char *)realloc(C,10*sizeof(char));

    free(A);
    free(B);
    free(C);
```

# Dinamično zasedanje pomnilnika – 2D

```
int B[3][4];
int **A;

int main(){
    A = (int **)malloc(3*sizeof(int *));
    for(int i = 0;i<3;i++)
        A[i] = (int *)malloc(4*sizeof(int));

    for(int i = 0;i<3;i++)
        free(A[i]);
    free(A);
}
```



# Naloga 1

Napiši program, ki vsebuje naslednje funkcije

- `double * Random(int n);`
  - Funkcija naj dinamično ustvari in vrne kazalec na polje `n` naključnih realnih števil med 0 in 1
- `double ** Matrix (double *A, int n, int r, int c);`
  - Funkcija naj dinamično ustvari matriko z r vrsticami in vanjo prepiše vrednosti iz polja A. Nato naj vrne kazalec na matriko. (Opomba: število stolpcev `c` morate določiti na podlagi števil `n` in `r`. Če je števil premalo dopolnite matriko z ničlami!)
- `double * Max(double *A, int n);`
  - Funkcija naj vrne kazalec na največjo vrednost v polju A
- Vse tri funkcije uporabite v glavnem programu, tako da polje, ki ga ustvari funkcija `Random` kot argument podate preostalima dvema funkcijama in izpišete rezultate.

Pomagajte si s spletno stranjo, kjer najdete vse potrebne funkcije:

- https://www.csse.uwa.edu.au/programming/ansic-library.html

# Naključna števila

```
#include <time.h>
#include <stdlib.h>

srand(time(NULL));

//celo število med 0 in RAND_MAX
int r = rand();
```

# Primer vhoda in izhoda

**Vhod**

```
Vnesi n: 8
Vnesi r: 2
```

**Izhod**

```
1D:
0.02 0.03 0.23 0.89 0.10 0.56 0.99 0.73
2D:
0.02 0.03 0.23 0.89
0.10 0.56 0.99 0.73

Najvecja vrednost: 0.99 na naslovu: 0012FEDC.
```

# Naloga 1

Rok za oddajo:

- 4. 11. 2021
- Zadnji možni (ena točka manj) 11. 11. 2021