«ԾՐԱԳՐԱՎՈՐՄԱՆ ՀԻՄՈՒՆՔՆԵՐ» դասընթաց

այլ ոլորտներից դեպի տեխնոլոգիական ոլորտ սկսնակների համար

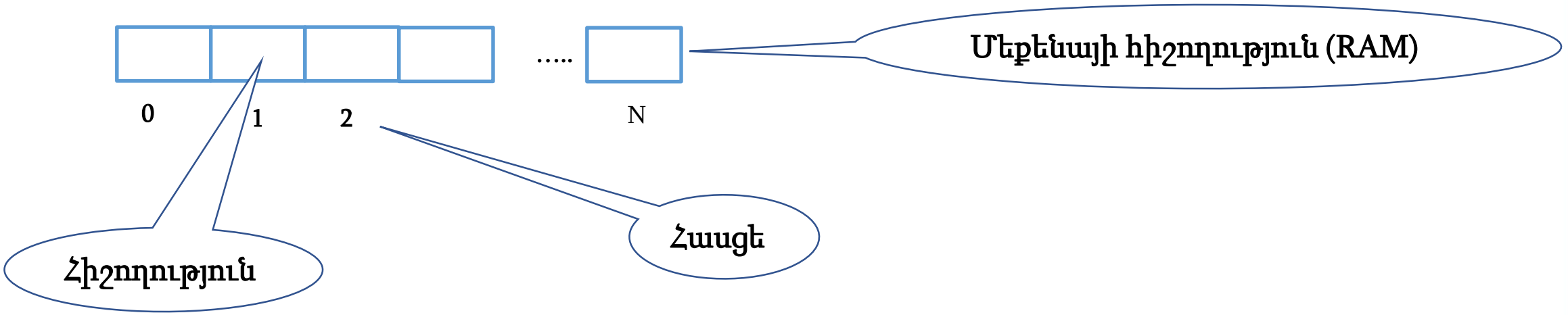
edu2020.am

ԴԱՍ #9

ՀԱՅ-ՌՈՒՍԱԿԱՆ ՀԱՄԱԼՍԱՐԱՆ

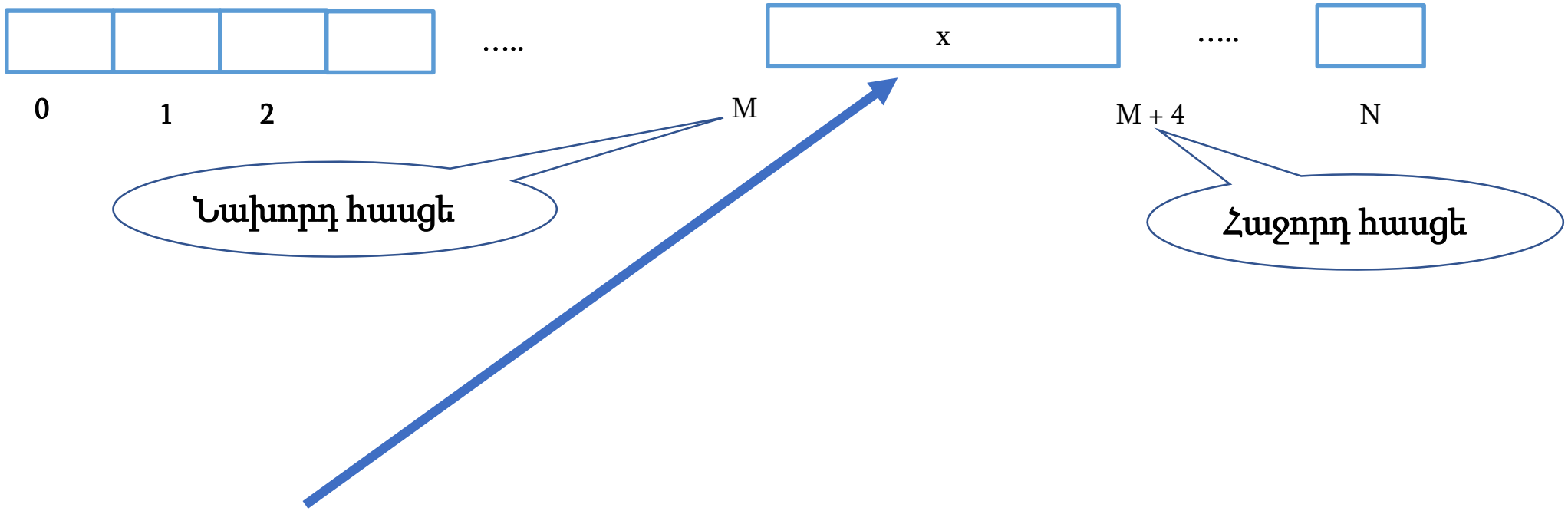
ՀԱՅԱՍՏԱՆԻ ՀԱՆՐԱՊԵՏՈՒԹՅԱՆ
ԲԱՐՁՐ ՏԵԽՆՈԼՈԳԻԱԿԱՆ
ԱՐԴՅՈՒՆԱԲԵՐՈՒԹՅԱՆ ՆԱԽԱՐԱՐՈՒԹՅՈՒՆ

Փոփոխականի հասցե



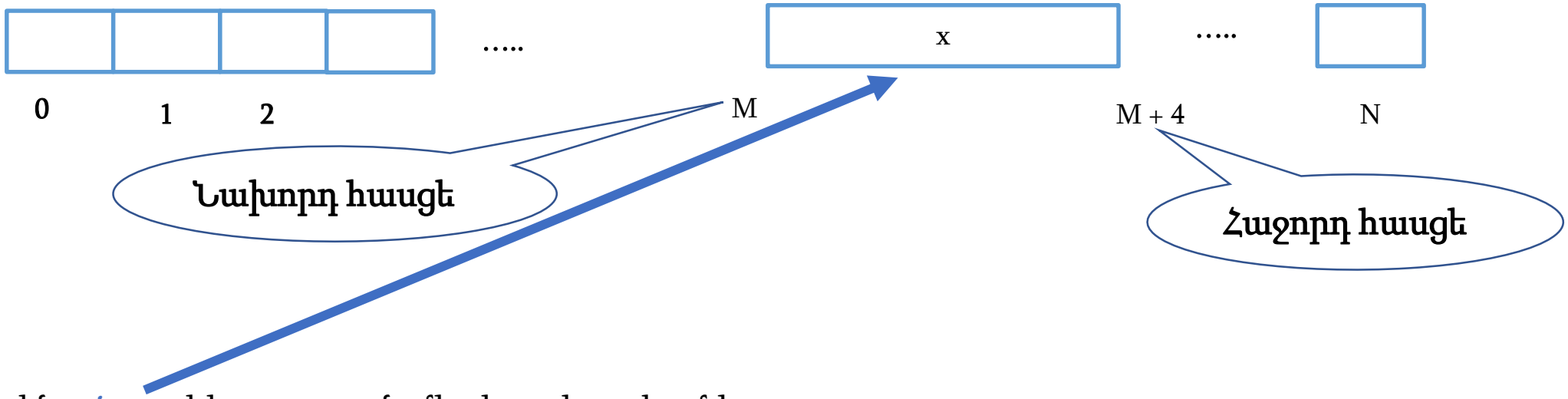
Հիշողության ամեն մի բայթ (byte) ունի իր ունիկալ հասցեն (համարակալված են)

Փոփոխականի հասցե



Եթե գրում ենք *int x ;* հիշողության մեջ հատկացվում է տեղ *x* փոփոխականի համար (*ենթադրենք int-ը 4 բայթ է*)

Փոփոխականի հասցե



Եթե գրում ենք *int x ;* հիշողության մեջ հատկացվում է տեղ *x* փոփոխականի համար (*ենթադրենք int-ը 4 բայթ է*)

C++ թույլ է տալիս ստանալ այդ փոփոխականի հասցեն **& (հասցեավորման օպերատոր) միջոցով**

& վերադարձնում է փոփոխականի հասցեն

```
#include <iostream>

int main() {
  int x;
  std::cout << "Address of x is " << &x << std::endl;
}
```

https://repl.it/@SevakRAU/ReferenceEx1

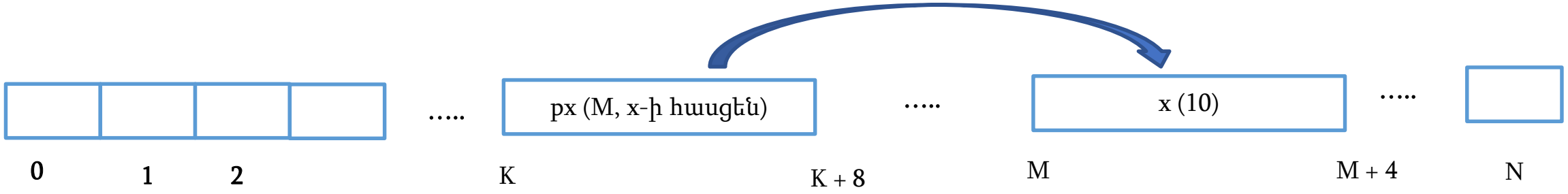
Ցուցիչներ

Ցուցիչը փոփոխական է, որը պարունակում է մեկ այլ փոփոխականի հասցե՝

```
int x = 10;
int* px = &x; // px–ի մեջ գրվում է M, որը x փոփոխականի արժեքն է
```

հասցեն է

Այլ կերպ ասում ենք, որ px-ը ցուցիչ է x-ի վրա՝ ցույց է տալիս հիշողության մեջ x-ի տեղը



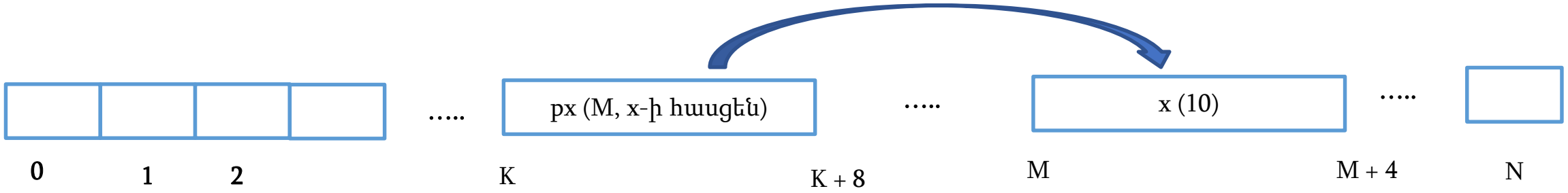
Ցուցիչներ

Ցուցիչը փոփոխական է, որը պարունակում է մեկ այլ փոփոխականի հասցե՝

int x = 10;

int px = &x; // px–ի մեջ գրվում է M, որը x փոփոխականի ~~արժեքն է~~ հասցեն է*

Այլապես ասում ենք որ px-ը ցուցիչ է x-ի վրա՝ ցույց է տալիս հիշողության մեջ x-ի տեղը



Ցուցիչը փոփոխական է՝ որը ևս պահվում է հիշողության մեջ, և կարելի է իր հասցեն էլ ստանալ
& (հասցեավորման օպերատոր) միջոցով

```
#include <iostream>
int main() {
  int x = 10;
  int *px = &x;
  std::cout << "Address of px is " << &px << std::endl;
} https://repl.it/@SevakRAU/PoniterRefEx1
```

Ցուցիչներ

```
#include <iostream>

int main() {
  int x = 10;
  int* px = &x;
  std::cout << &x << std::endl;

  std::cout << px << std::endl;
}
```

&x – x փոփոխականի հասցեն

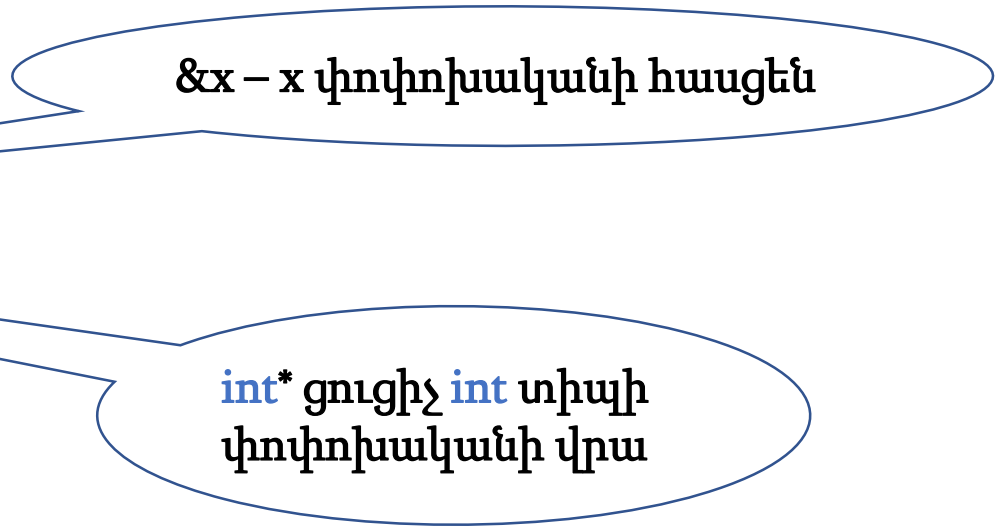
https://repl.it/@VahagVardanyan/SimplePointer

Ցուցիչներ

```
#include <iostream>

int main() {
  int x = 10;
  int* px = &x;
  std::cout << &x << std::endl;

  std::cout << px << std::endl;
}
```



https://repl.it/@VahagVardanyan/SimplePointer

int* p; համարժեք գրելաձևն է int *p;

Ցուցիչներ տարբեր տիպերի վրա

Ցուցիչ հայտարարելիս պետք է հաշվի առնել համապատասխան տիպը որ վրա ցույց է տալու:

```
#include <iostream>

int main() {
  int *ip;    // ցուցիչ int տիպի վրա
  double *dp; // ցուցիչ double տիպի վրա
  float *fp;  // ցուցիչ float տիպի վրա
  char *ch;   // ցուցիչ char տիպի վրա
}
```

Ցուցիչների ապահասցեավորում

Ապահասցեավորում (dereference) – վերցնել հասցեի պարունակությունը

```
#include <iostream>

int main() {

  int x = 10;
  int* px = &x;
  std::cout << px << std::endl;
  std::cout << *px << std::endl;
}
```

https://repl.it/@SevakRAU/PtrDerefEx1

Ցուցիչի չափ

Ցուցիչների չափը ցուցիչի տիպից կախված չէ:
Այն կախված է մեքենայի ճարտարապետությունից (բիթայնությունից):
32 բիթանոց համակարգիչներում 4 բայթ է (32 բիթ): 64 բիթանոց համակարգիչներում 8 բայթ է (64 բիթ):

```
#include <iostream>
int main() {
   int *ip; // ցուցիչ int տիպի վրա
   double *dp; // ցուցիչ double տիպի վրա
   float *fp; // ցուցիչ float տիպի վրա
   char *ch; // ցուցիչ char տիպի վրա


   std::cout << sizeof(ip) << std::endl;
   std::cout << sizeof(dp) << std::endl;
   std::cout << sizeof(fp) << std::endl;
   std::cout << sizeof(ch) << std::endl;
}
```

https://repl.it/@SevakRAU/SizeOfPtr

Գործողություններ ցուցիչների հետ

Ցուցիչների համար սահմանված են հետևյալ գործողությունները

- ++
- --
- +=
- -=
- +
- -
- >, >=
- <, <=
- ==
- !=

Գործողություններ ցուցիչների հետ

Ցուցիչների համար սահմանված են հետևյալ գործողությունները

- ++ *(int *p; p++ համարժեք է p=p + sizeof(int))*
- -- *(int *p; p-- համարժեք է p=p - sizeof(int))*
- += *(int *p; p+=i համարժեք է p=p + i*sizeof(int))*
- -= *(int *p; p-=i համարժեք է p=p – i*sizeof(int))*
- + *(int *p; p + i համարժեք է p + i * sizeof(int))*
- - *(int *p; p - i համարժեք է p - i * sizeof(int))*
- >, >= *(համեմատում է հասցեները)*
- <, <= *(համեմատում է հասցեները)*
- == *(համեմատում է հասցեները)*
- != *(համեմատում է հասցեները)*

Գործողություններ ցուցիչների հետ

```
#include <iostream>
int main() {
  int a = 10;
  int *p = &a;
  int *p1 = p, *p2 = p, *p3 = p;
  p1++;
  p2 += 4;
  p3 -= 1;
  std::cout << "size of int is " << sizeof(int) << std::endl;
  std::cout << " p is " << p << std::endl;
  std::cout << " p1 is p + sizeof(int)" << p1 << std::endl;
  std::cout << " p2 is p + 4 * sizeof(int)" << p2 << std::endl;
  std::cout << " p3 is p - sizeof(int)" << p3 << std::endl;
}
```

https://repl.it/@SevakRAU/PointerArithEx1

Գործողություններ ցուցիչների հետ

```
#include <iostream>

int main() {
  int a[10];
  int *p1 = &a[0], *p2 = &a[9];
  std::cout << "p1 " << p1 <<std::endl;
  std::cout << "p2 " << p2 <<std::endl;
  std::cout << (p1 > p2) << std::endl;
  std::cout << (p1 >= p2) << std::endl;
  std::cout << (p1 < p2) << std::endl;
  std::cout << (p1 <= p2) << std::endl;
}
```

https://repl.it/@SevakRAU/PointerCmpEx1

Գործողություններ ցուցիչների հետ

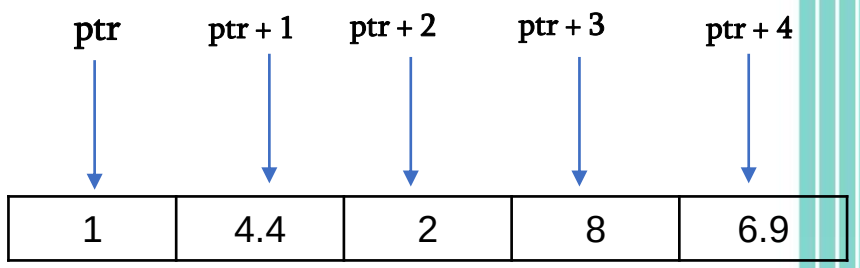
```
#include <iostream>
using namespace std;
int main() {
   float arr[5] = {1, 4.4, 2, 8, 6.9};
   float *ptr;

   cout << "Displaying address using arrays: " << endl;
   for (int i = 0; i < 5; ++i) {
     cout << "&arr[" << i << "] = " << &arr[i] << endl;
   }

   ptr = &arr[0];
   // ptr = arr;

   cout<<"\nDisplaying address using pointers: "<< endl;
   for (int i = 0; i < 5; ++i) {
     cout << "ptr + " << i << " = "<< ptr + i << endl;
   }

}
```



https://repl.it/@VahagVardanyan/arrayPointer

Գործողություններ ցուցիչների հետ

```
#include <iostream>
using namespace std;
int main() {
   float arr[5] = {1, 4.4, 2, 8, 6.9};
   float *ptr;

   cout << "Displaying address using arrays: " << endl;
   for (int i = 0; i < 5; ++i) {
     cout << "&arr[" << i << "] = " << &arr[i] << endl;
   }

   ptr = &arr[0];
   // ptr = arr;

   cout<<"\nDisplaying address using pointers: "<< endl;
   for (int i = 0; i < 5; ++i) {
     cout << "ptr + " << i << " = "<< ptr + i << endl;
   }

}
```

Տպել arr[i] – ի հասցեն

https://repl.it/@VahagVardanyan/arrayPointer

Գործողություններ ցուցիչների հետ

```
#include <iostream>
using namespace std;
int main() {
   float arr[5] = {1, 4.4, 2, 8, 6.9};
   float *ptr;

   cout << "Displaying address using arrays: " << endl;
   for (int i = 0; i < 5; ++i) {
     cout << "&arr[" << i << "] = " << &arr[i] << endl;
   }

   ptr = &arr[0];
   // ptr = arr;

   cout<<"\nDisplaying address using pointers: "<< endl;
   for (int i = 0; i < 5; ++i) {
     cout << "ptr + " << i << " = "<< ptr + i << endl;
   }

}
```

https://repl.it/@VahagVardanyan/arrayPointer

Տպել arr[i] – ի հասցեն

ptr պարունակում է զանգվածի առաջին էլեմենտի հասցեն

Ցուցիչների և զանգվածների կապը

```
#include <iostream>
using namespace std;
int main() {
   float arr[5] = {1, 4.4, 2, 8, 6.9};
   float *ptr;

   cout << "Displaying address using arrays: " << endl;
   for (int i = 0; i < 5; ++i) {
     cout << "&arr[" << i << "] = " << &arr[i] << endl;
   }

   ptr = &arr[0];
   // ptr = arr;

   cout<<"\nDisplaying address using pointers: "<< endl;
   for (int i = 0; i < 5; ++i) {
     cout << "ptr + " << i << " = "<< ptr + i << endl;
   }

}
```

https://repl.it/@VahagVardanyan/arrayPointer

Տպել arr[i] – ի հասցեն

ptr պարունակում է զանգվածի առաջին էլեմենտի հասցեն

i = 0: ptr + 0 == &a[0]
i = 1: ptr + 1 == &a[1]
i = 4: ptr + 4 == &a[4]

Ցուցիչների և զանգվածների կապը

Տպել զանգվածի էլեմենտները

```
#include <iostream>
using namespace std;
int main() {
   float arr[5] = {1, 4.4, 2, 8, 6.9};
   float *ptr;

   // ptr = &arr[0];
   ptr = arr;

   cout<<"\nDisplaying address using pointers: "<< endl;
   for (int i = 0; i < 5; ++i) {
     cout << ”*(ptr + " << i << ”) = "<< *(ptr + i) << endl;
   }

}
```

https://repl.it/@VahagVardanyan/arrayPointerV

Զանգվածի անունը նրա առաջին էլեմենտի հասցեն է՝ arr == &arr[0]

i = 0: *(ptr + 0) == a[0]
i = 4: *(ptr + 4) == a[4]

Խնդիրներ

1. Շրջանցել զանգված հակառակ կարգով՝ ունենալով ցուցիչ վերջին էլեմենտի վրա և իմանալով զանգվածի չափը:
2. Շրջանցել զանգվածի կենտ/զույգ ինդեքսների վրայով ունենալով ցուցիչ առաջին էլեմենտի վրա և իմանալով զանգվածի չափը:

Ցուցիչների սկզբնարժեքավորում

```
int* p;
int x = 10;
p = &x;
```

```
int *p = nullptr;
int x = 10;
p = &x;
```

Ցուցիչների սկզբնարժեքավորում

```
#include <iostream>
int main() {
  int *p = nullptr;
  std::cout << p <<std::endl;

  int x = 10;
  p = &x;

  std::cout << p <<std::endl;

}
```

nullptr – սկզբնարժեքավորել ցուցիչը 0 - ով

https://repl.it/@VahagVardanyan/BigOnerlookedCurrencies

Հղումներ

- Հղումը դա գոյություն ունեցող փոփխականի ալտերնատիվ անունն է (հիշողության մեջ առանձին տեղ չի հատկացվում)
- Փոփոխականը որպես հղում հայտարարելու համար պետք է փոփոխականի անունից առաջ դնել & սիմվոլը

```
int x = 10;
int& ref = x;
```

ref – փոփոխականը հղում է x փոփոխականի վրա: ref և x փոփոխականները նույն հիշողության տարբեր անուններն են

Հղումներ

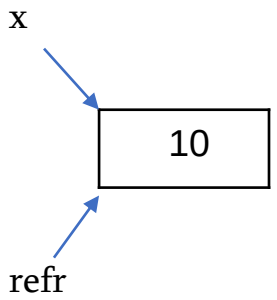
```
#include <iostream>

int main() {
  int x = 10;
  int& refr = x;

  x = 100;

  std::cout << x << std::endl;

  std::cout << refr << std::endl;
}
```



https://repl.it/@SevakRAU/RefEx1

Հղումների սկզբնարժեքավորում

- Հղումները հարկավոր է սկզբնարժեքավորել հայտարարման ժամանակ:

```
#include <iostream>

int main() {
  int x = 10;
  int& refr = x;

  int& invalidRefr;
  invalidRefr = &x;
}
```

https://repl.it/@SevakRAU/RefNoInit

Հղումների և ցուցիչների տարբերություն

1. Սկզբարժեքավորում
2. Ցուցիչներին հնարավոր է վերագրել նոր արժեք, հղումները ոչ
3. Ցուցիչների համար հատկացվում է հիշողություն, հղումների համար ոչ

Աշխատանք տողերի հետ

Տողը կարելի է ներկայացնել որպես սիմվոլների զանգված

```
char str [] = "Hello world";
```

Տողերը վերջանում են հատուկ սիմվոլով '\0'

Աշխատանք տողերի հետ

```
#include <iostream>

int main() {
  char str[2] = "ab"; // Compile error

  const int length = 10;
  char str2[length];
  std::cin >> str2;
  std::cout << str2;
}
```

https://repl.it/@VahagVardanyan/Chars

Կոմպիլացիայի սխալ: Պետք է հաշվի առնել, որ տողերը վերջանում են “\0” – ով: Զանգվածի չափը պետք է լինի 3

Աշխատանք տողերի հետ

```
#include <iostream>

int main() {
  char str[3] = "ab"; // OK

  const int length = 10;
  char str2[length];
  std::cin >> str2;
  std::cout << str2;
}
```

Տնային աշխատանք

Տնային աշխատանք 1-5:

Վարժություններ

- 00 Նախավարժանք
- 01 Թվաբանություն և ճյուղավորում
- 02 Ցիկլեր և ստատիկ զանգվածներ
- 03 Դինամիկ զանգվածներ և ֆունկցիաներ
- 04 Դասեր

Շնորհակալություն. Հարցե՞ր