«ԾՐԱԳՐԱՎՈՐՄԱՆ ՀԻՄՈՒՆՔՆԵՐ» դասընթաց

այլ ոլորտներից դեպի տեխնոլոգիական ոլորտ սկսնակների համար

edu2020.am

ԴԱՍ #11

ՀԱՅ-ՌՈՒՍԱԿԱՆ ՀԱՄԱԼՍԱՐԱՆ

ՀԱՅԱՍՏԱՆԻ ՀԱՆՐԱՊԵՏՈՒԹՅԱՆ
ԲԱՐՁՐ ՏԵԽՆՈԼՈԳԻԱԿԱՆ
ԱՐԴՅՈՒՆԱԲԵՐՈՒԹՅԱՆ ՆԱԽԱՐԱՐՈՒԹՅՈՒՆ

Ֆունկցիաներ

Ֆունկցիա – հրամանների հաջորդականություն, որը կարելի է կանչել –ծրագրի ցանկացած կետից

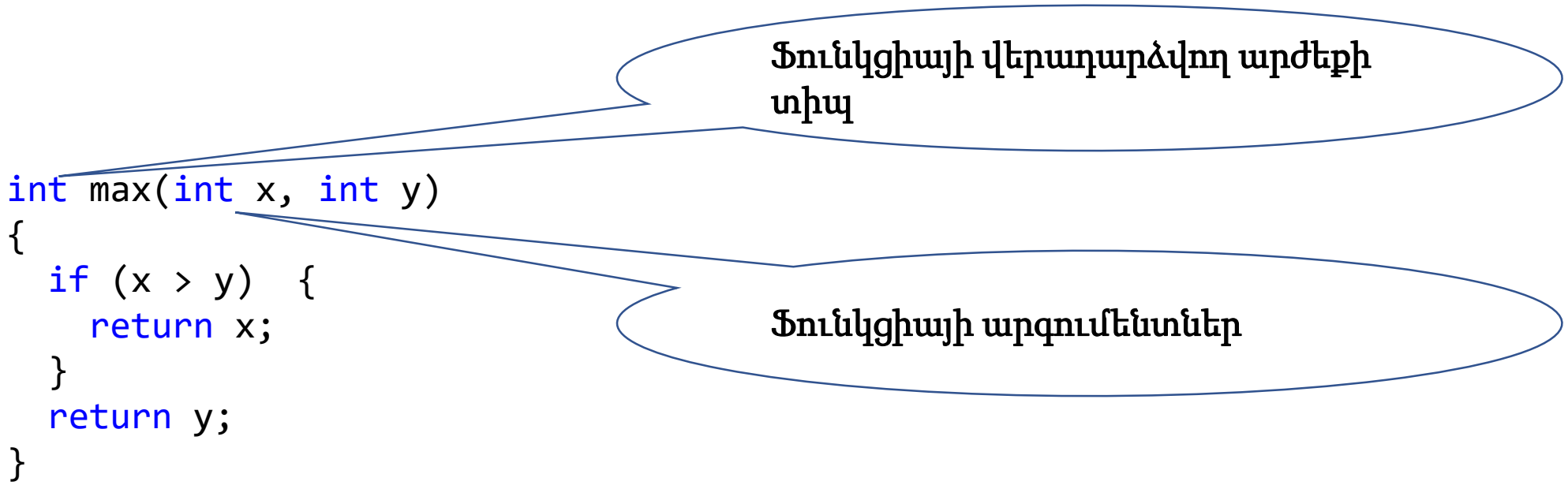
Նպատակ - գրելով կոդի կրկնվող հատվածները ֆունկցիաների մեջ կարող ենք խուսափել կոդի կրկնություններից

Օրինակ – կոդի հատվածը, որը պատասխանատու է զանգվածի սորտավորման համար կարելի է սահմանել որպես ֆունկցիա:

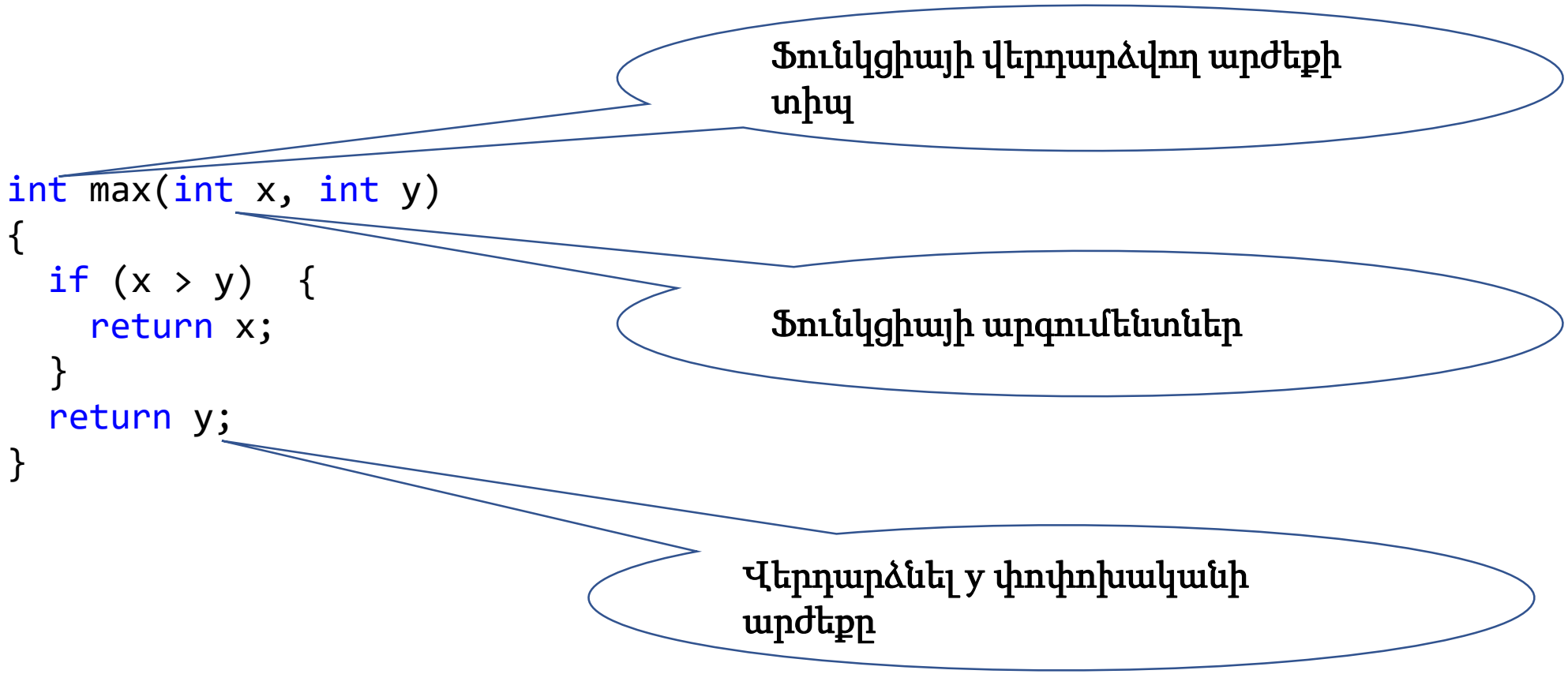
Ֆունկցիայի օրինակ

```
int max(int x, int y)
{
  if (x > y)  {
    return x;
  }
  return y;
}
```

Ֆունկցիայի օրինակ



Ֆունկցիայի օրինակ



Ֆունկցիաներ

```
return_type function_name( argument_list ) {
      body of the function
}
```

return_type – վերադարձվող արժեքի տիպ (int, short, float, double …)

function_name – ֆունկցիայի անուն, որը իրենից ներկայացնում է իդենտիֆիկատոր

argument_list – ֆունկցիայի արգումենտների ցուցակ (ֆունկցիայի մուտքային տվյալներ):

Ֆունկցիայի օրինակ

Շրջանի մակերես հաշվող ֆունկցիա:

```
double square(double radius) {
  return 3.14 * radius * radius;
}
```

Ֆունկցիայի կանչ

```
#include <iostream>

double square(double radius) {
  return 3.14 * radius * radius;
}

int main() {
  double radius;
  std::cin >> radius;
  double result = square(radius);
  std::cout << "result is " << result << std::endl;
}
```

square ֆունկցիայի նկարագրություն

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

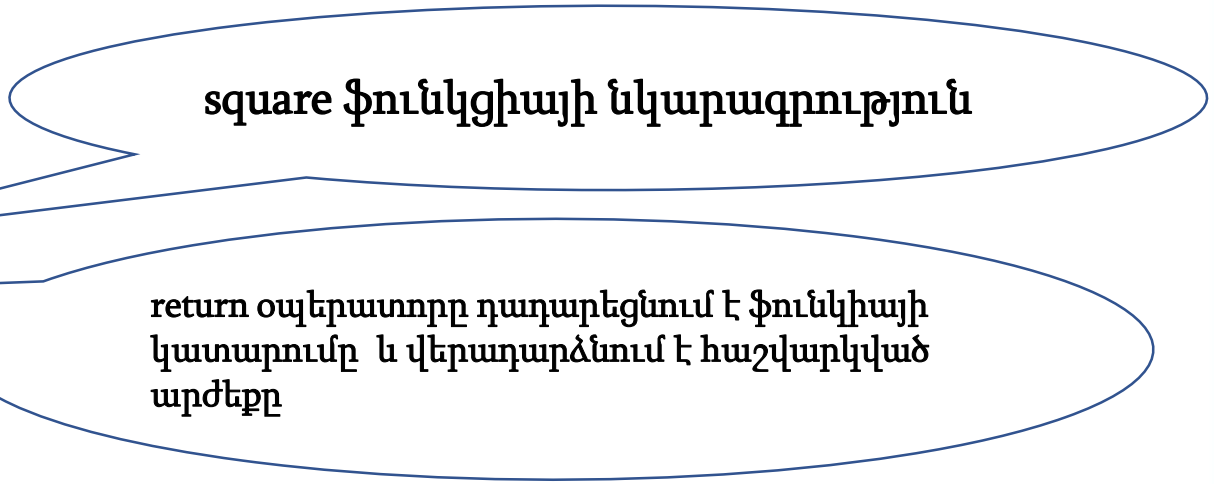
Ֆունկցիայի կանչ

```
#include <iostream>

double square(double radius) {
  return 3.14 * radius * radius;
}

int main() {
  double radius;
  std::cin >> radius;
  double result = square(radius);
  std::cout << "result is " << result << std::endl;
}
```

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



Ֆունկցիայի կանչ

```
#include <iostream>

double square(double radius) {
  return 3.14 * radius * radius;
}

int main() {
  double radius;
  std::cin >> radius;
  double result = square(radius);
  std::cout << "result is " << result << std
}
```

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

square ֆունկցիայի նկարագրություն

return օպերատորը դադարեցնում է ֆունկիայի կատարումը և վերադարձնում է հաշվարկված արժեքը

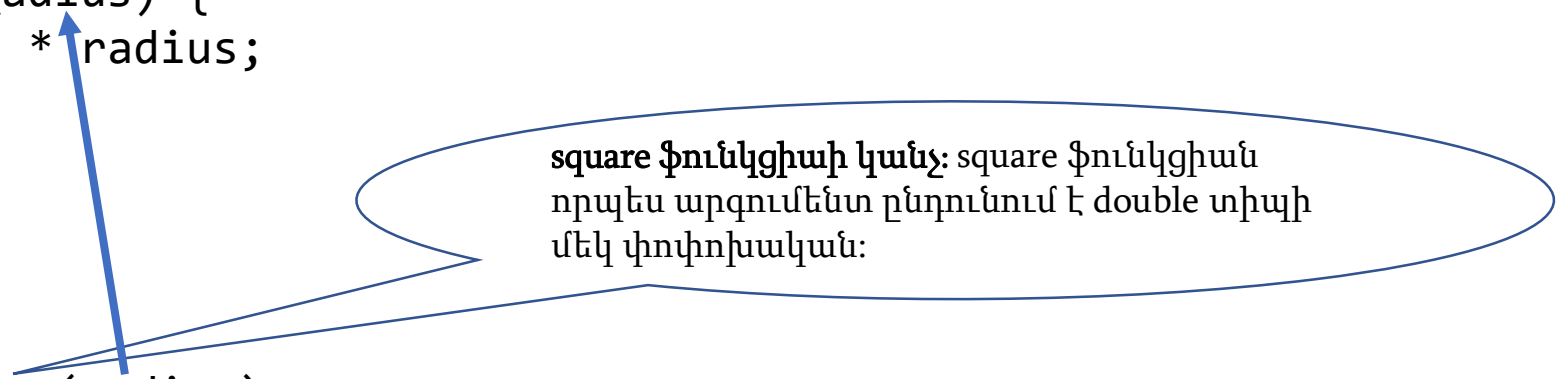
square ֆունկցիան որպես արգումենտ ընդունում է *double* տիպի մեկ փոփոխական

Ֆունկցիայի կանչ

```
#include <iostream>

double square(double radius) {
  return 3.14 * radius * radius;
}

int main() {
  double radius;
  std::cin >> radius;
  double result = square(radius);
  std::cout << "result is " << result << std::endl;
}
```



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

Ֆունկցիայի կանչ

```
#include <iostream>

double square(double radius) {
  return 3.14 * radius * radius;
}

int main() {
  double radius;
  std::cin >> radius;
  double result = square(radius);
  std::cout << "result is " << result << std::endl;
}
```

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

Ֆունկցիայի արգումենտների և փոխանցվող փոփոխականների անունները կարող են չհամընկնել

square ֆունկցիաի կանչ: square ֆունկցիան որպես արգումենտ ընդունում է double տիպի մեկ փոփոխական:

Ֆունկցիայի կանչ

```
#include <iostream>

double square(double radius) {
  return 3.14 * radius * radius;
}

int main() {
  double radius;
  std::cin >> radius;
  double result = square(radius);
  std::cout << "result is
}
```

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

square ֆունկցիայի կանչ: square ֆունկցիան որպես արգումենտ ընդունում է double տիպի մեկ փոփոխական:

square ֆունկցիայի կանչ: square ֆունկցիան վերադարձնում է double տիպի արժեք:

Ֆունկցիայի տեսանելիություն

```
#include <iostream>

int main() {
  double radius;
  std::cin >> radius;
  double result = square(radius);
  std::cout << "result is " << result << std::endl;
}

double square(double radius) {
  return 3.14 * radius * radius;
}
```

Թարգմանության սխալ

Ֆունկցիան տեսանելի է իր հայտարարությունից հետո գտնվող տողերում միայն

Ֆունկցիայի տեսանելիություն

Ֆունկցիայի հայտարարություն:
Տեղեկացնում ենք որ ծրագրում պետք է գոյություն ունենա նմանատիպ ֆունկցիա

```
#include <iostream>

double square(double radius);
// double square(double); // կարելի է radius անունը բաց թողնել

int main() {
  double radius;
  std::cin >> radius;
  double result = square(radius);
  std::cout << "result is " << result << std::endl;
}

double square(double radius) {
  return 3.14 * radius * radius;
}
```

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

return օպերատոր

- Ֆունկցիան վերջանում է return հրամանով:
- Ֆունկցիան կարող է վերադարձնել միայն մեկ արժեք
- Ֆունկցիան կարող է վերադարձնել ցանկացած տիպ բացի զանգված և ֆունկցիա տիպերից:

```
#include <iostream>

double square(double radius) {
  if (radius < 0) {
    std::cout << "Wrong radius"<< std::endl;
    return 0;
  }
  return 3.14 * radius * radius;
}
```

return օպերատորը դադարեցնում է ֆունկիայի կատարումը և վերադարձնում է հաշվարկված արժեքը:

Եթե շառավիղը փոքր է 0 – ից դադարացնել ֆունկցիայի աշխատանքը և վերադարձնել 0

void տիպ

Բոլոր ֆունկցիաները ունեն վերադարձվող արժեքի տիպ։

Երբեմն հարկավոր է գրել ֆունկցիաներ, որոնք ոչ մի բան չեն վերադարձնում։ Այդ դեպքերում օգտագործվում է void տիպը

```
#include <iostream>

void prettyPrint(int input) {
  std::cout << input << std::endl;
}

int main() {
  int a = 10;
  int b = a + 1;
  prettyPrint(a);
  prettyPrint(b);
}
```

Տպել մուտքային արգումենտը և անցնել հաջորդ տող

void տիպ

Բոլոր ֆունկցիաները ունեն վերադարձվող արժեքի տիպ։

Երբեմն հարկավոր է գրել ֆունկցիաներ, որոնք ոչ մի բան չեն վերադարձնում։ Այդ դեպքերում օգտագործվում է void տիպը

```
#include <iostream>

void prettyPrint(int input) {
  std::cout << input << std::endl;
  return;
}

int main() {
  int a = 10;
  int b = a + 1;
  prettyPrint(a);
  prettyPrint(b);
}
```

Տպել մուտքային արգումենտը և անցնել հաջորդ տող

`void` վերադարձվող արժեքի դեպքում `return` կարելի է գրել կամ չգրել

Խնդիրներ

1. Գրել ֆունկցիա, որը որպես արգումենտներ ստանում է ուղղանկյան կողմերը և վերադարձնում դրա մակերեսը/պարագիծը:
2. Գրել ֆունկցիա, որը որպես արգումենտ ստանում է N դրական թիվը և վերադարձնում 1-ից N թվերի գումարը:

Արգումենտների փոխանցում: Առանց արգումենտների ֆունկցիա

```
#include <iostream>

void printHello() {
  std::cout << "Hello" << std::endl;
}

int main() {
  printHello();
}
```

Ֆունկցիան կարող է լինել առանց արգումենտների

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



Արգումենտների փոխանցում ըստ արժեքի

Արդեն դիտարկված օրինակներում արգումենտները փոխանցվում են ֆունկցիային ըստ արժեքի:

```
#include <iostream>

void testArg(double arg1, int arg2) {
  arg1 = arg1 + 1;
  arg2 = 0;
  std::cout << "Inside function" << std::endl;
  std::cout << arg1 << " " << arg2 << std::endl;
}

int main() {
  double a = 4.3;
  int b = 5;
  testArg(a, b);
  std::cout << "After function" << std::endl;
  std::cout << a << " " << b << std::endl;
}
```

a և b փոփոխականների արժեքները պատճենվում են նոր հիշողության մեջ:

a և b փոփոխականների արժեքների փոփոխությունը ֆունկցիայի մեջ չի ազդում նրանց արժեքների վրա ֆունկցիայից դուրս

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

Արգումենտների փոխանցում ըստ հղման

```
#include <iostream>
void testArg(double& arg1) {
  arg1 = arg1 + 1;
  std::cout << "Inside function" << std::endl;
  std::cout << arg1 << std::endl;
}

int main() {
  double a = 4.4;
  testArg(a);

  std::cout << "After function" << std::endl;
  std::cout << a << std::endl;
}
```

testArg ֆունկցիան ընդունում է որպես արգումենտ հղում double տիպի փոփոխականի վրա

Ֆունկցիայի կանչի ժամանակ arg1 անունով հղումը սկզբնարժեքավորվում է a փոփոխականի արժեքով

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

Արգումենտների փոխանցում ըստ ցուցիչի

```
#include <iostream>
void testArg(double* arg1) {
  arg1 = arg1 + 1; // Աշխատանք հասցեի հետ
  std::cout << "Inside function" << std::endl;
  std::cout << arg1 << std::endl;
}

int main() {
  double a = 4.4;
  testArg(&a);
  testArg(nullptr);
  std::cout << "After function" << std::endl;
  std::cout << a << std::endl;
}
```

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

testArg ֆունկցիան ընդունում է որպես արգումենտ double տիպի փոփոխականի հասցե

Ֆունկցիայի կանչի ժամանակ փոխանցվում է a փոփոխականի հասցեն

Թույլատրվում փոխանցել 0/nullptr

Արգումենտների փոխանցում ըստ ցուցիչի

```
#include <iostream>
void testArg(double* arg1) {
  *arg1 = *arg1 + 1;
  std::cout << "Inside function" << std::endl;
  std::cout << *arg1 << std::endl;
}
int main() {
  double a = 4.4;
  testArg(&a);
  //testArg(nullptr);
  std::cout << "After function" << std::endl;
  std::cout << a << std::endl;
}
```

Աշխատանք անմիջապես փոփոխականի արժեքի հետ

Այս դեպքում nullptr չի կարելի փոխանցել որովհետև testsArg ֆունկցիայում ապահասցեավորում է տեղի ունենում

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

Օրինակ՝ swap ֆունկցիա

```
#include <iostream>
void swap(int& x, int& y) {
  int z = x;
  x = y;
  y = z;
}
int main() {
  int a = 45, b = 35;
  std::cout << "Before Swap\n";
  std::cout << "a = " << a << " b = " << b << "\n";
  swap(a, b);
  std::cout << "After Swap with pass by reference\n";
  std::cout << "a = " << a << " b = " << b << "\n";
}
```

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

Խնդիրներ

1. Գրել ֆունկցիա, որը որպես արգումենտներ կստանա 3 թվեր և այդ թվերը կմեծացնի համապատասխանաբար 1-ով, 2-ով և 3-ով (արգումենտները փոխանցել հղումով/ցուցիչով):

Զանգվածը որպես ֆունկցիայի արգումենտ

```
#include <iostream>
void printArray(int a [], int size) {
  for (int i = 0; i < size; i++) {
    std::cout << a[i] << " ";
  }
  std::cout << std::endl;
}

void printArray2(int* a , int size) {
  for (int i = 0; i < size; i++) {
    std::cout << a[i] << " ";
  }
  std::cout << std::endl;
}

int main() {
  const int size = 5;
  int arr[size] = {1, 2, 3, 4, 5};
  printArray2(arr, size);
}
```

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



Օրինակ. BubleSort ֆունկցիա

```
#include <iostream>
void swap(int& a, int& b) {
  int tmp = a;
  a = b;
  b = tmp;
}
void BubleSort(int arr[], int size) {
  int i, j;
  bool swapped;
  for (i = 0; i < size - 1; i++) {
    swapped = false;
    for (j = 0; j < size - 1; j++) {
      if (arr[j] > arr[j+1]) {
        swap(arr[j], arr[j+1]);
        swapped = true;
      }
    }
    // IF no two elements were swapped by inner loop, then break
    if (swapped == false)
      break;
  }
}
```

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

Օրինակ. BubleSort ֆունկցիա

```
void printArray(int a[], int size) {
  for (int i = 0; i < size; i++) {
    std::cout << a[i] << " ";
  }
  std::cout << std::endl;
}

int main() {
  int arr[] = {8, 6, 3, 9};
  int arr2[] = {4, 2, 45, 5, 9, 3};
  BubleSort(arr, 4);
  BubleSort(arr2, 6);
  printArray(arr, 4);
  printArray(arr2, 6);
}
```

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

Խնդիրներ

1. Գրել ֆունկցիա, որը շրջում է զանգվածը (օգտագործել զանգված փոխանցելու [], * գրելաձևերը)։

Main ֆունկցիա

- Ամեն ծրագիր սկսում է իր աշխատանքը main ֆունկցիայից:
- main վերդարձնում է int տիպի փոփոխական (0 – ծրագրի հաջող ավարտի դեպքում)

```
#include <iostream>

int main() {
  double a = 4.4;
  std::cout << a << std::endl;
  return 0;
}
```

Ամփոփում

- Ֆունկցիաներ կարելի է օգտագործել կրկնություններից խուսափելու համար

- Կոդը դառնում է ավելի հասկանալի

Տնային աշխատանք

Տնային աշխատանք 16-21:

Վարժություններ

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

Շնորհակալություն. Հարցե՞ր