C++ ....SELFNOTES

Sunday, October 13 at 13:10 | 5532 characters

🔴🔴iostream🔴🔴

=It's a header file .contains functions for basic input and output operations..

🔴🔴#include iostream 🔴🔴

=Including that header file.

🔴🔴int main () {

return 0;

}🔴🔴

=While we are return 0 at the end of the main function,it's say's that there is no problems in this program.

🔴

Std::(standard)

cout(character output)
<<(it's also known as output)

<< std::endl; endline

<< '\n' ; newline

🔴we can use both endl; and /n for forming a newline.endline gives a better performance,endline will flush the output buffer .. really we can use either one.

🔵std::court << " I am BHAGATH" << std::endl;
std::cout << " I love her " << std::endl;

concluding

🔴At the end of the statement,we add a semicolon (;)

🔴//This is a comment 🔴

Comments are ignored by the complier

🔴/*

We

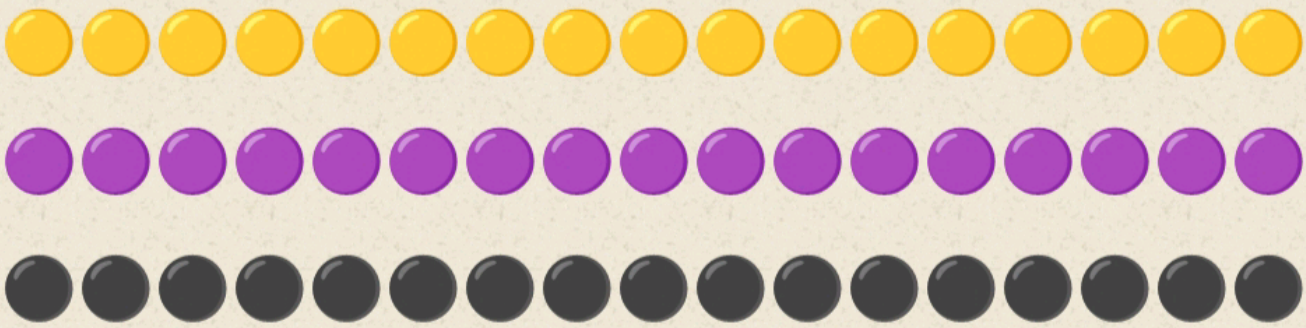
need

a

multi-line

comment

*/🔴



🔴 Variables 🔴

```
#include <iostream>


int main() {

        int x; // declaration
        x = 5; // assignment

        std::cout << x;

         return 0;
}
```

output

5

```
#include <iostream>


int main() {

       int x =5;
       int y =6;
       int sum = x+y;


       std::cout << x << '\n';
       std::cout << y << '\n';
       std::cout << sum << '\n';
```

```
    return 0;

}
```

output

```
5
6
11
```

🔴 while we are using integer, we can't print decimal portion

There we use ' double '(number including decimal portion)🔴

```
#include <iostream>

int main() {

double price : 20.99;
double gpa:2.8;
double Celsius:8.8;

std :: cout << gpa;

return 0;
}
```

output
2.8

🔴// single character 🔴

'Char'

```
char variable name = ' A ';

(variable names like grade.....)
```

Chars only store's a single character

🔴// Boolean (true or false)

```
bool variable name = false;
bool variable name=true;

(variable names like student, light,so on)
```

🔴//String (objects that represents a sequence of text)

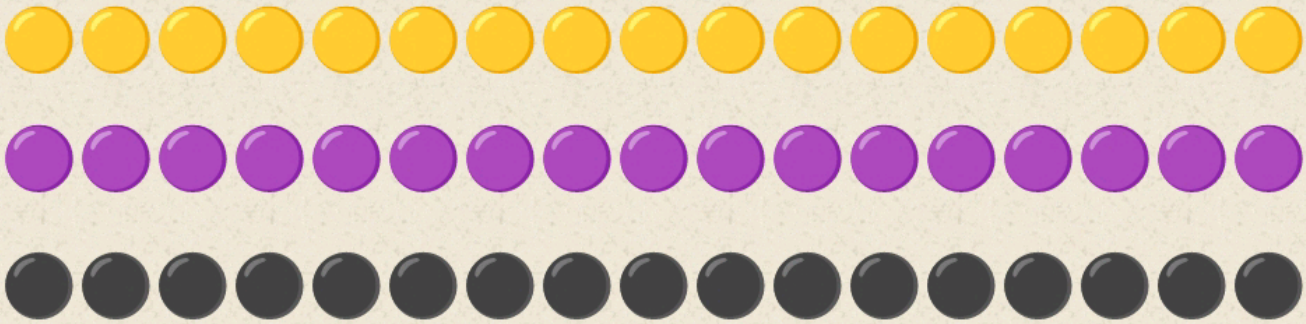
```
std:: string name = " Bhagat";
std:: string day = " Friday";

std::cout << name ;
```

output

Bhagath

```
std:: string name = " Bhagat";
std:: string day = " Friday";

std::cout <<" HELLO  " << name;
```

```
HELLO Bhagath
```



🔴🔴Const

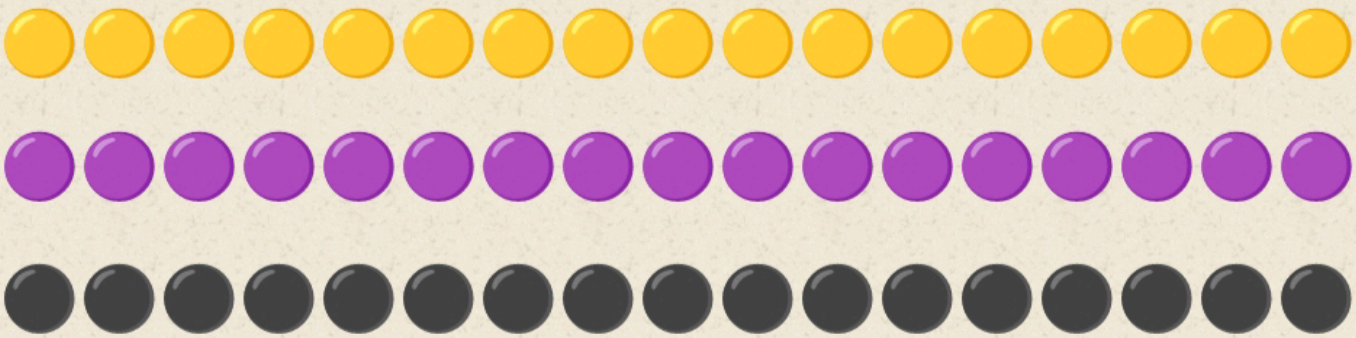
The const keyword specifices that a variable's value is constant.

Tells the complier to prevent anything from modifying it.

You know that a variable is not going to be changed at all ..then put a const....

```
#include <iostream>

int main() {

const double PI = 3.14159;
//PI = 4.20; THIS WILL CAUSE AN ERROR
double radius=10;
double circumference=2*PI*radius;

std::cout<<circumference <<"cm";

return 0;
}
```

```
const double PI= 3.14;
const int Width=10;
const int height=56;
const light_speed=299792458;
```



🔴Namespace

While we are "using namespace std;"

We don't need to add std:: in the entire program

```
#include <iostream>

int main() {
    using namespace std;

    string name = "bhagath";

    cout << "Hello "<< name;

    return 0;
}
```

```
Hello bhagath
```

🔴::

scope resolution operator

```
#include <iostream>

namespace first {
        int x=1;
}
namespace second {
        int x=2;
}

int main() {
        int x=0;
        std::cout<< x;
        return 0;
}
```

output
0

```
#include <iostream>

namespace first {
        int x=1;
}
namespace second {
        int x=2;
}

int main() {
        int x=0;
        std::cout<<first::x;
        return 0;
}
```

output
1

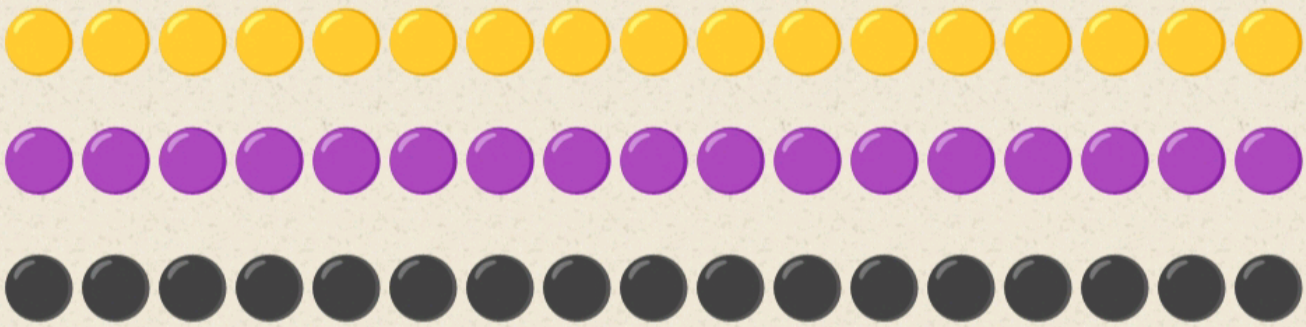
```
#include <iostream>
namespace first {
        int x=1;
}
namespace second {
        int x=2;
}
int main() {
      using namespace first;
      int x=0;
      std::cout<<x;
      return 0;
}
```

output
1

```
#include <iostream>
namespace first {
       int x=1;
}
namespace second {
      int x=2;
}
```

```
}
int main() {
	using namespace first;
	int x=0;
	std::cout<<second::x;
	return 0;
}
```

output
2



🔴Typedef

Reserved keyword used to create an additional name (alias) for another data type.

🟢Typedef keyword the new identifier usually ends with underscore t(_t)

```
#include <iostream>
#include <vector>

typedef std::string text_t;
typedef int number_t ;

int main(){

   text_t firstName = "Bro";
   number_t age = 21;
```

```
std:: cout << firstName <<'\n';
std:: cout << age <<'\n';

return 0;
}
```

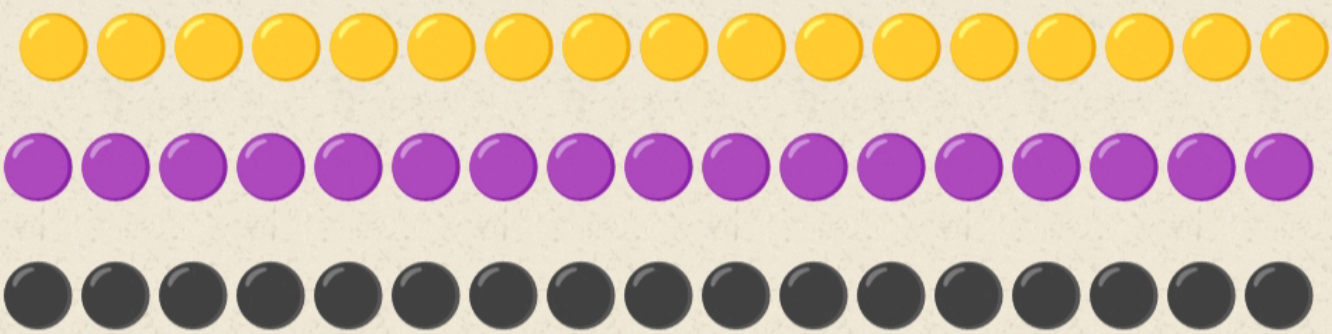
```
Bro
21
```

🔴Using

```
#include <iostream>
#include <vector>

using text_t = std::string;
using number_t = int ;

int main(){

   text_t firstName = "Bro";
   number_t age = 21;
```

```
std:: cout << firstName <<'\n';
std:: cout << age <<'\n';

return 0;
}
```

```
Bro
21
```



🔴 Arithmetic operators

int students=20;

(if we want decimal,Then use double)

🟢 students=students+1;
🟢 students+=1;
🟢 students++;

output
21

🟢 students=students-1;
🟢 students-=1;
🟢 students--;

output
19

🟢 students=students*2;
🟢 students*=2;

output 40

🟢 students=students/2;
🟢 students/=2;

output 10

🔴

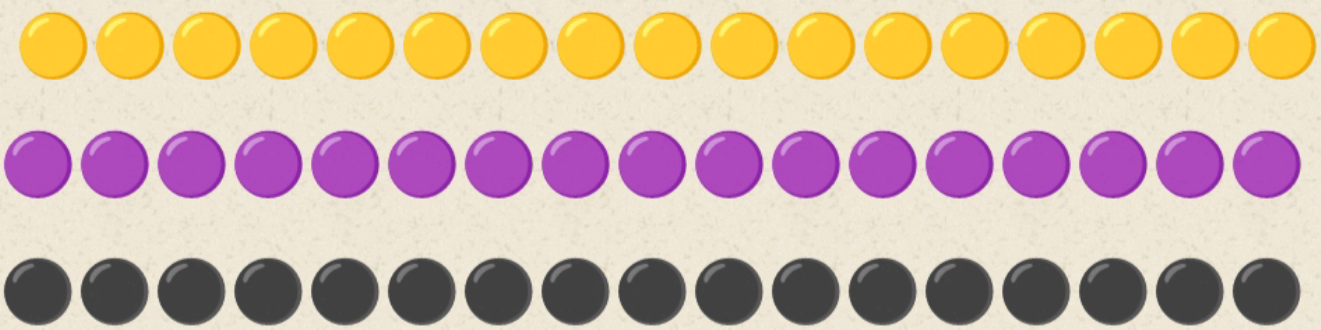
```
int students=3+4*2/6
```

```
output (muli,div,add,sub)

4.3
```

🔴if we want the remainder

Use %

```
int students

int remainder= students % 3




....
.....
....
...
```



🔴Type conversation

Implicit/explicit

```
double x = (int) 3.14;

std::cout<< x;
```

3

🟢

```
int correct=7
int questions=10
double score = correct/(double) questions*100;

std::cout << score << "%";

return 0;
```

80%

here we don't use the double.. answer will give 0 because of the decimal

🔴getline

You need to accept a string that includes any white spaces you're better off using the getline function.

((((((((((((Std::ws

This portion will eliminate any newline character or any white spaces before any user input)))))))))

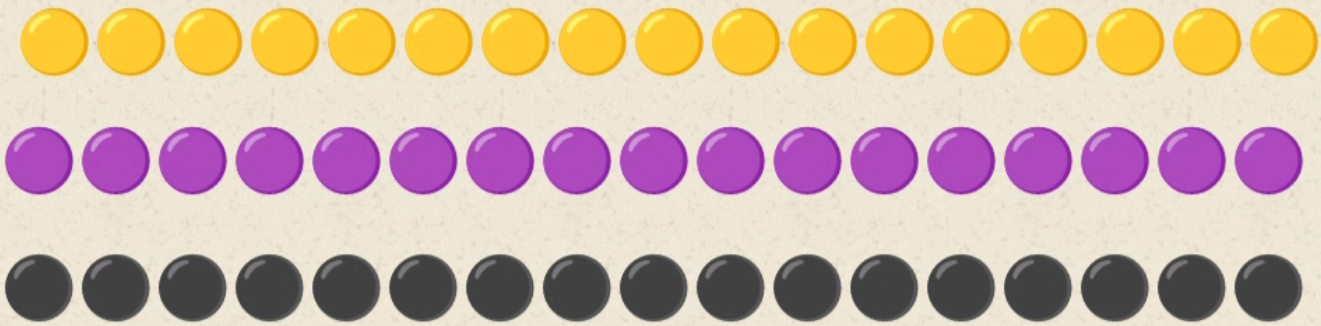
```
cout<< (insertion operator)
cin<<( extraction operator)
```

```
int main(){
      std::string name;
      int age;

      std::cout<<"What's your age?:";
      std::cin>>age;
```

```
std::cout<<"What's your full name?:";
std::getline(std::cin>>std::ws,name);

std::cout << "Hello "<< name << '\n';
std::cout << " You are " << age << "  years old ";
```

output

What's your age?:18
What's your full name?: Bhagath Sen (🟢here we use space between bhagath sen.thats y we use getline function 🟢)
Hello Bhagath sen
You are 18 years old



🔴Math functions

```
#include <iostream>
#include<cmath>

int main()
{
    double x= 3;
    double y =4;
    double x;

 🟢 z=std::max(x ,y);
 🟢 z=std::min(x,y);
 🟢 z=pow(2,4);
 🟢 z=sqrt(9);
 🟢 z=abs(-3);
 ...
```

output

```
4
3
16
3
3
```

```
#include <iostream>
#include<cmath>

int main()
{
    double x= 3.14;
    double y =4;
    double x;

    z= round(x);
    z=ceil(x);
```

```
3
4
```

```
#include <iostream>
#include<cmath>

int main()
{
    double x= 3.99;
    double y =4;
    double x;

    z=floor(x);

    std::cout << z;

    return 0;
}
```

```
3
```