Practicum woensdag: pointers en references

Collapse context

Voor de automatische verbetering op INGinious is het belangrijk dat je na elke cout ook een endl toevoegt!

Your answer passed the tests! Your score is 100.0%. [Submission #6734e5616c698110d4c1c312 (**2024-11-13 17:44:01**)]

Question 1: Integer Division

✓ Perfect

Schrijf een functie `integerDivision`. Deze functie krijgt als input een teller en noemer (beide ints). De functie returnt true als de deling opgaand is, dat wil zeggen als de rest van de deling gelijk is aan 0. Geef quotiënt en rest als extra parameter by reference aan de functie mee om deze te kunnen doorgeven.

```
1  #include <iostream>
2  using namespace std;
3
4  int integerDivision(int teller, int noemer, int &quotient, int& rest)
5  {
6      rest = teller % noemer;
7      quotient = teller / noemer;
8
9      return rest == 0;
10 }
11
12
13
```

Submit

Question 2: Optellen met pointers

✓ Perfect

Schrijf een functie `add(a,b,result)` die twee pointers naar een int (`a` en `b`) krijgt en een pointer naar een int `result` als derde parameter voor het resultaat. De functie geeft niets terug en zet de som in de variabele `result`.

```
1  void add(int *a,int *b,int *result)
2  {
3      *result = *a + *b;
4  }
```

Submit

Question 3: Compare by value

✓ Perfect

Schrijf een functie `compareByValue(p,i)` die een pointer naar een int `p` krijgt en een integer `i`. De functie geeft true terug als de pointer verwijst naar een waarde die gelijk is aan i, anders false.

```
int a = 5;
int* b = &a;
int c = 5;
int d = 3;
// boolalpha zal de waarde van de boolean afdrukken als true of false ipv 1 of 0
cout << boolalpha << compareByValue(b,a) << endl; // geeft true
cout << boolalpha << compareByValue(b,c) << endl; // geeft true
cout << boolalpha << compareByValue(b,d) << endl; // geeft false
```

```
1  bool compareByValue(int *p, int i)
2  {
3      return *p == i;
4  }
```

Submit

Question 4: Compare by address

✓ Perfect

Schrijf een functie `compareByAddress(p,i)` die een pointer naar een int `p` krijgt en een integer `i`. De functie geeft true terug als de pointer verwijst naar dezelfde geheugenplaats als die van i, anders false.

```
int a = 5;
int* b = &a;
int c = 5;
int d = 3;
// boolalpha zal de waarde van de boolean afdrukken als true of false ipv 1 of 0
cout << boolalpha << compareByAddress(b,a) << endl; // geeft true
cout << boolalpha << compareByAddress(b,c) << endl; // geeft false
cout << boolalpha << compareByAddress(b,d) << endl; // geeft false
```

```
1  bool compareByAddress(int *p, int &i)
2  {
3      return p == &i;
4  }
```

Submit

Question 5: Dobbelen

✓ Perfect

Schrijf een functie `throwDice(d1,d2)` om met twee dobbelstenen te gooien. De functie geeft niets terug en gebruikt de parameters om de nieuwe waarden van de dobbelstenen door te geven. Gebruik de functie `rand()` die een random integer genereert.

```
void throwDice(int& d1, int& d2)
{
    d1 = rand() % 6 + 1;
    d2 = rand() % 6 + 1;
}
```

Submit

Question 6: Shift

✓ Perfect

Schrijf een functie `shiftLeft(a,b,c)` die 3 getallen krijgt en ze circulair doorschuift, d.w.z. a krijgt de waarde van b, b die van c en c die van a. De functie geeft niets terug.

```
void shiftLeft(int& a, int& b, int& c)
{
    int temp = a;
    a = b;
    b = c;
    c = temp;
}
```

Submit

Question 7: Shift pointers

✓ Perfect

Schrijf een functie `shiftLeft(a,b,c)` die 3 pointers naar getallen krijgt en de waarden (niet de pointers!) circulair doorschuift. De functie geeft niets terug.

```
void shiftLeft(int* a, int* b, int* c)
{
    int temp = *a;
    *a = *b;
    *b = *c;
    *c = temp;
}
```

Submit

Question 8: Python print

✓ Perfect

We implementeren de `print` functie uit Python. Schrijf een functie `print(a,n)` die een array van karakters krijgt en een integer die de lengte van de array bevat. De functie print alle karakters achter elkaar (elk karakter afzonderlijk) en eindigt met een newline (enter). Denk eraan dat je een array doorgeeft door een pointer naar het eerste element. Je mag uiteraard geen `cout << a` gebruiken.

```
void print(char a[], int n)
{
    for (int i = 0; i < n; i++)
    {
        cout << *(a+i);
    }
}
```

Submit

Question 9: Python print zonder lengte

✓ Perfect

We implementeren opnieuw de `print` functie uit Python, maar deze keer geven we geen lengte mee. Schrijf een functie `print(a)` die een array van karakters krijgt. De functie print alle karakters achter elkaar en eindigt met een newline (enter). De array bestaat uit een aantal karakters gevolgd door het speciale karakter \0. Gebruik geen index of [] operator (dus geen a[i]), maar gebruik de ++ operator op pointers om vooruit te gaan.

```
void print(char* a)
{
    for (char* i = a; *i != '\0'; ++i)
    {
        cout << *i;
    }
}
```

Submit

Question 10: Klinkers tellen

✓ Perfect

Schrijf een functie `telKlinkers` die het aantal klinkers (a, e, i, o, u en dus niet de y) in een tekst telt. De functie krijgt een array van karakters mee en geeft het aantal klinkers terug (de array eindigt tevens met het speciale karakter \0). Gebruik geen strings of string functies en gebruik geen [] operator (dus geen a[i] bv), maar gebruik de ++ operator op pointers om vooruit te gaan.

```
int telKlinkers(char* txt)
{
    int result = 0;
    for (char* i = txt; *i != '\0'; ++i)
    {
        if (*i == 'a' || *i == 'e' || *i == 'i' || *i == 'o' || *i == 'u')
            result++;
    }
    return result;
}
```

Submit