**­­МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ**

**УЧРЕЖДЕНИЕ ОБРАЗОВАНИЯ**

**ГОМЕЛЬСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ**

**УНИВЕРСИТЕТ ИМЕНИ П. О. СУХОГО**

Факультет автоматизированных и информационных систем

Кафедра «Информационные технологии»

**ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №3**

по дисциплине «Разработка приложений для *iPhone* и *iPad*»

на тему: «Классы в языке Swift»

Выполнил: студент гр. ИТП-22

Расшивалов Н.И.

Принял: ст. преподаватель

Семенченя Т. С.

Гомель 2021

**Цель работы:** изучить основные понятия и методы объектно-ориентированного подхода в swift.

**Вариант 9.**

− Ввод с клавиатуры данных о сотрудниках предприятия в массив объектов класса Employee.

− Вывод списка всех сотрудников предприятия в порядке возрастания средней зарплаты.

− Определение максимальной зарплаты по каждому отделу.

− Вывод списка сотрудников, которые средняя зарплата которых превышает среднюю по предприятию больше чем на 5% (если таких сотрудников нет, вывести соответствующее сообщение).

Класс Employee должен содержать закрытые поля: фамилия, наименование отдела, зарплата за последние n месяцев (массив), сабскрипт для доступа к элементам массива зарплат и все необходимые для решения задачи свойства и методы.

**Результат выполнения:**

Главное меню программы представлено на рисунке 1:

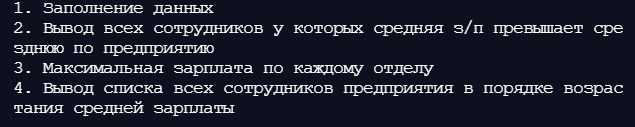


Рисунок 1 ­– Главное меню программы

Ввод с клавиатуры данных о сотрудниках предприятия в массив объектов класса *Employee* представлен на рисунке 2:

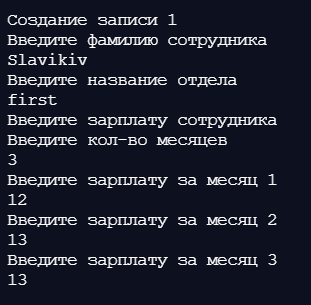


Рисунок 2 ­– Создание записи о сотрудниках

Вывод списка всех сотрудников предприятия в порядке возрастания средней зарплаты представлен на рисунке 3:

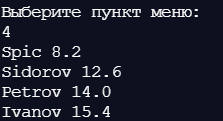


Рисунок 3 ­– вывод списка сотрудников

Метод определения максимальной зарплаты по каждому отделу представлен на рисунке 4:

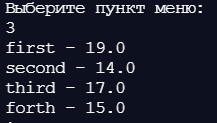


Рисунок 4 ­– Максимальная зарплата по отделам

Вывод списка сотрудников, у которых средняя зарплата превышает среднюю по предприятию больше чем на 5%, представлен на рисунке 5:

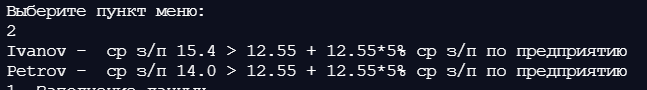


Рисунок 5 ­– Список сотрудников с искомой зарплатой

**Вывод:** в ходе выполнения лабораторной работы были изучены основные понятия и методы объектно-ориентированного подхода

**Листинг программы:**

Класс:

class Employee

{

var surname:String

var departmentName:String

var salary = [Double]()

init (surname: String, department: String, salary: [Double]) {

self.surname = surname

self.departmentName = department

self.salary = salary

}

subscript(index: Int) -> Double {

get {

return salary[index]

}

set(newValue) {

salary[index] = newValue

}

}

var length : Int {

get {

return salary.count

}

}

func getAverageSalary() -> Double {

var sum : Double = 0

for month in salary {

sum += month

}

return sum / Double(salary.count)

}

func getInfo(qua : Int) -> String {

var info = "\nФИО - " + surname + "\nОтдел - " + departmentName + "Зарплата - "

info += salary[0...qua - 1].description

return info

}

}

Программа:

class Employee

{

var surname:String

var departmentName:String

var salary = [Double]()

init (surname: String, department: String, salary: [Double]) {

self.surname = surname

self.departmentName = department

self.salary = salary

}

subscript(index: Int) -> Double {

get {

return salary[index]

}

set(newValue) {

salary[index] = newValue

}

}

var length : Int {

get {

return salary.count

}

}

func getMaxSalary() -> Double {

var max : Double = 0

for month in salary {

if max < month{

max = month

}

}

return max;

}

func getAverage() -> Double {

var sum : Double = 0

var average : Double = 0

for month in salary {

sum = sum + month

}

average = sum / Double(salary.count)

return average

}

func getInfo(qua : Int) -> String {

var info = "\nФИО - " + surname + "\nОтдел - " + departmentName + "Зарплата - "

info += salary[0...qua - 1].description

return info

}

}

**Листинг программы** *main():*

func Create(quantity: Int) -> [Employee] {

var employees = [Employee]()

for i in 0...quantity - 1 {

print("\nСоздание записи " + String(i + 1))

print("Введите фамилию сотрудника")

let name = readLine()!

print("Введите название отдела")

let depName = readLine()!

print("Введите зарплату сотрудника")

print("Введите кол-во месяцев")

let n = readLine()!

var salary = [Double]()

for j in 0...Int(n)! - 1 {

print("Введите зарплату за месяц " + String(j + 1))

salary.append(Double(readLine()!)!)

}

let employee: Employee = Employee(surname:name, department:depName, salary:salary)

employees.append(employee)

}

return employees

}

func AverageSalary(employees : [Employee]) {

var allAverage:Double = 0

for employee in employees{

allAverage = allAverage + employee.getAverage()

}

allAverage = allAverage / Double(employees.count)

for employee in employees{

if employee.getAverage() > allAverage + allAverage\*0.05{

print(employee.surname + " - " + " ср з/п " + String(employee.getAverage()) + " > " + String(allAverage) + " + " + String(allAverage) + "\*5%" + " ср з/п по предприятию" )

}

}

}

func MaxSalary(employees : [Employee]) {

var departments = [String]()

departments.append(employees[0].departmentName)

for employee in employees {

var flag = true

for department in departments {

if employee.departmentName == department {

flag = false

}

}

if flag {

departments.append(employee.departmentName)

}

}

var maxSalary = [Double]()

for department in departments {

for employee in employees {

if department == employee.departmentName {

maxSalary.append(employee.getMaxSalary())

}

}

}

var i = 0

for department in departments {

print(department + " - " + String(maxSalary[i]))

i += 1

}

}

func GetInfo(\_ employeess : [Employee]) {

var i = 0

var temp:Employee

var employees:[Employee] = employeess

while i < employees.count-1

{

var j = 0

while j < employees.count - i - 1

{

if employees[j].getAverage() > employees[j+1].getAverage()

{

temp = employees[j]

employees[j] = employees[j+1]

employees[j+1] = temp

}

j += 1

}

i += 1

}

for employee in employees{

print("\(employee.surname) \(employee.getAverage())")

}

}

var flag = true

var employes = [Employee]()

employes.append(Employee(surname:"Ivanov", department:"first", salary:[19,13, 14, 15, 16]))

employes.append(Employee(surname:"Sidorov", department:"second", salary:[13,12, 12, 14, 12]))

employes.append(Employee(surname:"Petrov", department:"third", salary:[17, 12, 12, 14, 15]))

employes.append(Employee(surname:"Spic", department:"forth", salary:[0, 0, 12, 14, 15]))

while flag {

print ("1. Заполнение данных")

print ("2. Вывод всех сотрудников у которых средняя з/п превышает срезднюю по предприятию")

print ("3. Максимальная зарплата по каждому отделу")

print ("4. Вывод списка всех сотрудников предприятия в порядке возрастания средней зарплаты")

print ("\nВыберите пункт меню:")

let item = readLine()!

switch Int(item)! {

case 1:

print("Введите кол-во новых записей")

let n = readLine()!

employes = Create(quantity: Int(n)!)

case 2:

AverageSalary(employees:employes)

case 3:

MaxSalary(employees:employes)

case 4:

GetInfo(employes)

case 5:

flag = false

default:

print("Неверный пункт")

}

}