Министерство образования Республики Беларусь

Учреждение образования

«Брестский государственный технический университет»

Факультет электронно-информационных систем

Кафедра интеллектуальных информационных технологий

ОТЧЕТ

о прохождении ознакомительной практики

с 06.07 по 19.07 2022 г.

Студента факультета электронно-информационных систем

(название факультета)

1 курс, ПО-9 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ А. С. Мисиюк

(курс, группа) (подпись)

Руководитель практики

от университета

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Т. А. Глущенко

(должность) (подпись)

13 страниц.

Отчёт ставит перед собой цель доказать факт прохождения ознакомительной практики, показать полученные знания и опыт при посещении предприятий и лабораторий, а также предъявить доказательства выполнения практических и индивидуального задания. В связи с этим отчет содержит следующие разделы:

* Титульный лист.
* Реферат.
* Содержание.
* Введение.
* Основная часть.
  + Задания
* Заключение.
* Список использованных источников.
* Приложения.

# Содержание

[Содержание 3](#_Toc109130680)

[Введение 4](#_Toc109130681)

[Основная часть 5](#_Toc109130682)

[ООО Харброс солюшинс 5](#_Toc109130683)

[Отраслевая лаборатория «Системы идентификации и промышленная робототехника» 5](#_Toc109130684)

[ООО "Эполь Софт" 5](#_Toc109130685)

[Научная лаборатория «Адаптивные системы управления транспортом» 6](#_Toc109130686)

[Задания 7](#_Toc109130687)

[Битовая маска 7](#_Toc109130688)

[Генерация перестановок 7](#_Toc109130689)

[Индивидуальное задание 7](#_Toc109130690)

[Заключение 9](#_Toc109130691)

[Список использованных источников 10](#_Toc109130692)

[Приложения 11](#_Toc109130693)

# Введение

Цель практики – дать студентам наиболее полное представление о будущей профессии и ее места в отрасли, объяснить взаимосвязь и направленность изучаемых по специальности дисциплин, закрепить на практике и расширить знания, полученные студентами при работе с персональным компьютером.

Задачи практики:

* ознакомление со структурой административного и оперативного управления предприятиями Беларуси (на примере выбранных для практики), ознакомление с его вычислительным центром;
* ознакомление с практическим состоянием дел в области развития информационных технологий на предприятии, также возможно предложить свои идеи по развитию.

В результате прохождения ознакомительной практики должны сформироваться базовые академические и профессиональные компетенции, в которые могут включаться навыки работы и обмена информацией в команде.

# Основная часть

### ООО Харброс солюшинс

HARBROS – молодая и активно растущая компания, занимающаяся в основном разработкой мобильных приложений под IOS. На момент прохождения практики в Бресте существует 2 офиса, в штате около 70 человек, в компании планируют и дальше набирать студентов и обучать их.

В компании используют язык swift – надёжный и интуитивно понятный язык программирования от Apple, при помощи которого можно создавать приложения для iOS, Mac, Apple TV и Apple Watch.

Для развития компании я бы рекомендовал развивать направление в сторону разработки приложений под Andriod, чтобы не упускать возможность разрабатывать приложения для клиента на большинстве мобильных устройств.

### Отраслевая лаборатория «Системы идентификации и промышленная робототехника»

Лаборатория на базе университета занимается практическими заданиями и спонсируется крупными предприятиями, позволяя студентам попробовать себя в профессиональной работе с роботами и соответствующим оборудованием. Также, тут, в случае заинтересованных лиц, можно получить трудоустройство.

Отраслевая лаборатория робототехники имеет несколько проектов, которые в основном являются работой с роботами, автоматизируя процесс на предприятии. Например маркировка товаров.

Лаборатория имеет большое количество оборудования, а также роботов. Одним из известных примеров будет робот-соперник для игры в шахматы или роботизированный кукольный театр.

### ООО "Эполь Софт"

Эполь Софт – ИТ-компания с 15-летним опытом работы на международном рынке. Предоставляют комплексные услуги по разработке, поддержке программного обеспечения и аутсорсингу для зарубежных и отечественных заказчиков.

Используют довольно широкий ассортимент языков программирования таких, как C, C++, Java, Javascript, Python и другие.

Для студента легко устроиться в эту фирму, необходимо лишь сдать тесты на логику, язык программирования и английский язык. После есть вариант с стандартной стажировкой или же лабораторией, где предоставляется возможность сразу работать в команде.

### Научная лаборатория «Адаптивные системы управления транспортом»

Основной задачей, которую решает лаборатория, является разработка теоретических основ оптимального управления автотранспортом в улично-дорожной сети города, внедрение полученных результатов и доведение опытных образцов изделий до серийного выпуска.

Лаборатория занимается такими проектами, как автоматические склады, адаптивные светофоры, “умная” система общественного транспорта и другими. Это довольно интересные и полезные для общества идеи, которые можно реализовать в нашем городе.

## Задания

### Битовая маска

Битовая маска — набор битов (обычно в числе), которые используются для маскирования — выбора отдельных битов или полей из нескольких битов из двоичной строки или числа.

Для кодирования множеств, каждому элементу множества соответсвует отдельный бит.

Операции над множествами реализуются с помощью стандартных побитовых опреациях

Код программы смотреть в *Приложении А*.

### Генерация перестановок

Код программы для генерации перестановок в лексикографическом порядке смотреть в *Приложении Б*.

Порядок следования перестановок в *антилексикографическом* порядке для перестановки из 3 элементов 1, 2, 3 в порядке возрастания:

**321 312 231 213 132 123**

### Индивидуальное задание

(Проблемы многопоточности)

Многопоточность — свойство (напр. приложения), состоящее в том, что процесс, порождённый в операционной системе, может состоять из нескольких потоков, выполняющихся «параллельно», то есть без предписанного порядка во времени.

Когда потоки должны взаимодействовать друг с другом или работать с общими данными, могут возникать проблемы.

#### Состояние гонки (race condition)

Состояние гонки — ошибка проектирования многопоточной системы или приложения, при которой работа системы или приложения зависит от того, в каком порядке выполняются части кода.

Состояние гонки — «плавающая» ошибка, проявляющаяся в случайные моменты времени и «пропадающая» при попытке её локализовать.

#### Взаимная блокировка (deadlock)

Deadlock — ситуация, при которой несколько потоков находятся в состоянии ожидания ресурсов, занятых друг другом, и ни один из них не может продолжать выполнение. Такое поведение может встречаться в неправильно спроектированных базах данных.

#### Голодание потоков

Голодание потоков — это ситуация, в которой поток не может получить доступ к общим ресурсам, потому что на эти ресурсы всегда претендуют какие-то другие потоки, которым отдаётся предпочтение.

#### Livelock

Поток часто действует в ответ на действие другого потока. Если действие другого потока также является ответом на действие первого потока, то может возникнуть **livelock**. Потоки не блокируются — они просто слишком заняты, реагируя на действия друг друга, чтобы возобновить работу.

# Заключение

Во время прохождения ознакомительной практики было получено представление о профессии, условиях трудоустройства и работы в офисах и лабораториях. Полученны базовые навыки для работы в области информационных технологий.

В составе группы мы посетили 2 предприятия, посетили лабораторию робототехники и транспортных систем на базе университета. Полученная информация является ценным ресурсом для будущего развития в этой области.

Получен практический опыт работы с множествами и решения задач на множества на языках программирования C и C++.

# Список использованных источников

* *Epol Soft — Belarusian IT-company*. https://www.epolsoft.com/ru/home\_ru/
* *Multithreading / Хабр*. Хабр: https://habr.com/ru/company/otus/blog/549814/
* *Swift – Apple (RU)*. https://www.apple.com/ru/swift/
* *Адаптивные системы управления транспортом - БрГТУ*. БрГТУ: https://www.bstu.by/nauka/nirs/innovatsionnye-razrabotki/adaptivnye-sistemy-upravleniya-transportom
* *Отраслевая лаборатория «Системы идентификации и промышленная робототехника» - БрГТУ*. БрГТУ: https://www.bstu.by/nauka/uslugi-tt/otraslevaya-lab-sistemy-identifikatsii-i-prom-robototekhnika
* *Работа в компании Харброс солюшинс* . https://rabota.by/employer/3887843

# Приложения

Приложение А

#include <iostream>

using namespace std;

int input\_universum[] = {1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11};

const int input\_universum\_size = 11;

int input\_A[] = {1, 2, 3, 4, 5, 6}; const int input\_A\_size = 6;

int input\_B[] = {5, 6, 7, 8}; const int input\_B\_size = 4;

int input\_C[] = {2, 3, 5, 7, 8}; const int input\_C\_size = 5;

//отображение в двоичном представлении

void convert\_to\_bin (unsigned decimal\_value);

//конвертация в битовую маску

int convert\_to\_bit\_mask(const int\* set, const int set\_size);

//булеан (функция возвращает битовые маски)

int\* create\_boolean(const int\* set, const int set\_size);

//объединение, пересечение, разность, симметрическую разность (функции возвращают битовые маски)

int unite(const int\* set1, const int set1\_size, const int\* set2, const int set2\_size);

int intersection(const int\* set1, const int set1\_size, const int\* set2, const int set2\_size);

int difference(const int\* set1, const int set1\_size, const int\* set2, const int set2\_size);

int symmetrical\_difference(const int\* set1, const int set1\_size, const int\* set2, const int set2\_size);

int addition(const int\* set1, const int set1\_size, const int\* set2, const int set2\_size);

int main()

{

int boolean\_size = 1 << (input\_A\_size);

int\* boolean = create\_boolean(input\_A, input\_A\_size);

for (int i = 0; i < boolean\_size; i++) {

convert\_to\_bin(boolean[i]);

cout << endl;

}

}

void convert\_to\_bin (unsigned decimal\_value)

{

char digit = (decimal\_value % 2) ? '1' : '0';

unsigned next = decimal\_value / 2;

if (next)

{

convert\_to\_bin (next);

}

printf("%c", digit);

}

int convert\_to\_bit\_mask(const int\* set, const int set\_size) {

int bitmask = 0;

for (int j = 0; j < set\_size; j++) {

bitmask += 1 << j;

}

return bitmask;

}

int\* create\_boolean(const int\* set, const int set\_size) {

int bitmask = convert\_to\_bit\_mask(set, set\_size);

int boolean\_size = 1 << (set\_size);

int\* boolean = new int[boolean\_size];

for (int i = 0; i < boolean\_size; i++) {

boolean[i] = 0;

}

int bm = bitmask;

int n = 0;

while (bm > 0) {

bm = bm >> 1;

n++;

}

int k = 1;

for (int j = 0; j <= n; j++) {

int bit = 1 << j;

int d = 1 << k;

if (bitmask%2) {

k++;

for (int i = 0; i < boolean\_size; i++) {

if (i%d < d/2) boolean[i] += bit;

}

}

bitmask = bitmask >> 1;

}

return boolean;

}

int unite(const int\* set1, const int set1\_size, const int\* set2, const int set2\_size) {

int bitmask1 = convert\_to\_bit\_mask(set1, set1\_size);

int bitmask2 = convert\_to\_bit\_mask(set2, set2\_size);

return bitmask1 | bitmask2;

}

int intersection(const int\* set1, const int set1\_size, const int\* set2, const int set2\_size) {

int bitmask1 = convert\_to\_bit\_mask(set1, set1\_size);

int bitmask2 = convert\_to\_bit\_mask(set2, set2\_size);

return bitmask1 & bitmask2;

}

int difference(const int\* set1, const int set1\_size, const int\* set2, const int set2\_size) {

int bitmask1 = convert\_to\_bit\_mask(set1, set1\_size);

int bitmask2 = convert\_to\_bit\_mask(set2, set2\_size);

return bitmask1 & (~(bitmask1 & bitmask2));

}

int symmetrical\_difference(const int\* set1, const int set1\_size, const int\* set2, const int set2\_size) {

int bitmask1 = convert\_to\_bit\_mask(set1, set1\_size);

int bitmask2 = convert\_to\_bit\_mask(set2, set2\_size);

return (bitmask1 & (~(bitmask1 & bitmask2))) | (bitmask2 & (~(bitmask1 & bitmask2)));

}

int addition(const int\* set1, const int set1\_size, const int\* set2, const int set2\_size) {

int bitmask1 = convert\_to\_bit\_mask(set1, set1\_size);

int bitmask2 = convert\_to\_bit\_mask(set2, set2\_size);

return bitmask1 & (~(bitmask1 & bitmask2));

}

Приложение Б

#include <iostream>

using namespace std;

int main()

{

int\* elements;

int kolvo;

cout << "Enter amount of elements\n";

cin >> kolvo;

elements = (int\*) malloc(kolvo \* sizeof(int));

kolvo--;

for(int i = 0; i <= kolvo; i++) {

elements[i]=i+1;

cout << elements[i] << " ";

}

cout << endl;

for(int i = kolvo; i > 0; i--) {

if(elements[i] > elements[i-1]) {

int idx = i;

for(i = kolvo; i >= idx; i--) {

if(elements[i] > elements[idx-1]) {

int element = elements[idx-1];

elements[idx-1] = elements[i];

elements[i] = element;

for(int j = kolvo, i = idx; j > i; j--, i++) {

element = elements[i];

elements[i] = elements[j];

elements[j] = element;

}

for(i=0;i<=kolvo;i++) cout << elements[i] << " ";

cout << endl;

break;

}

}

}

}

return 0;

}