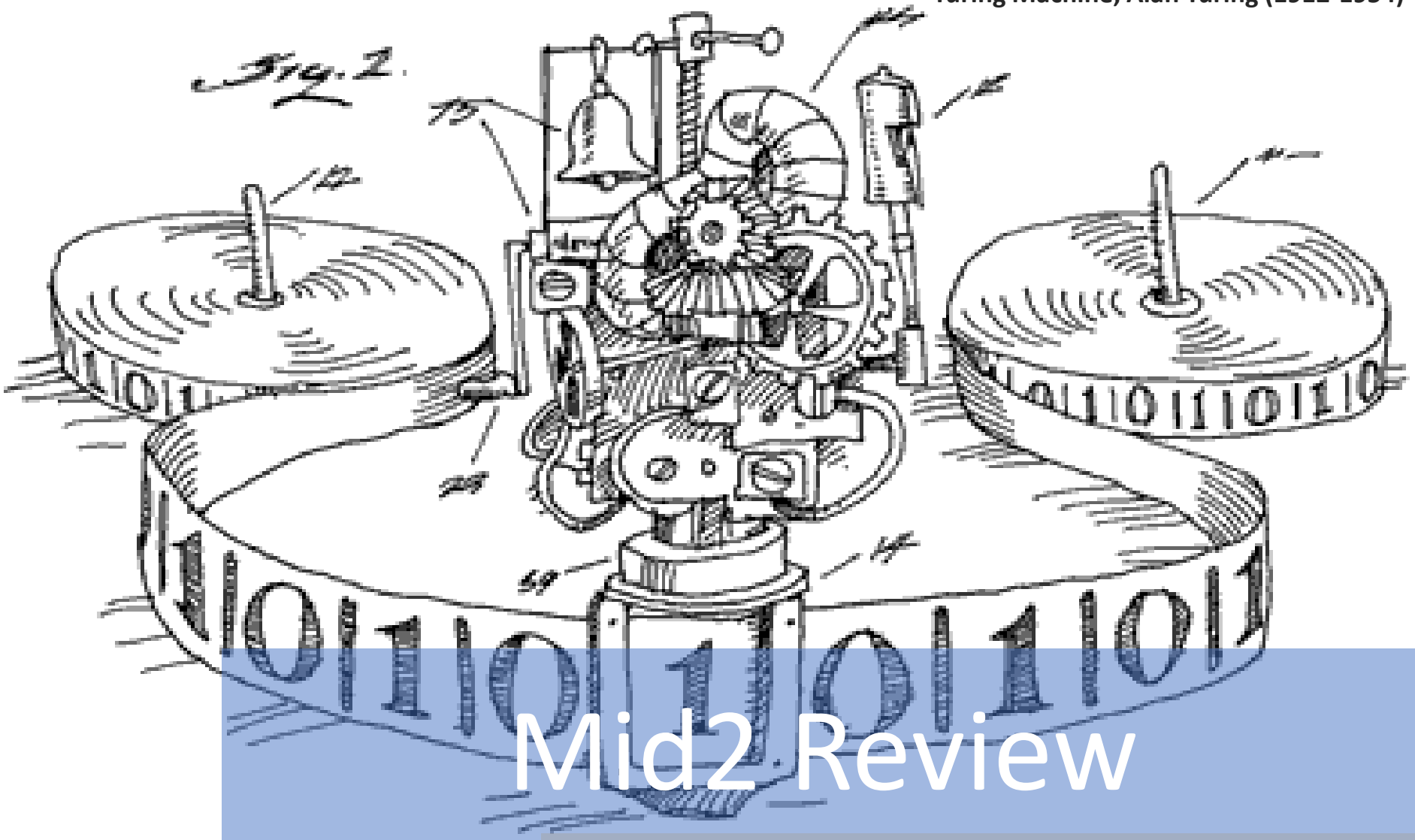
Turing Machine, Alan Turing (1912-1954)



Mid2 Review

VG101 TA Group / Patrick Yao

Yao Yue 2016.18

?

How to use this document?

Copy selected notes on to your lecture notes

You are allowed to bring annotated lecture notes into exam.

Use as a check list for your knowledge!

Make sure you understand everything.

Re: ZERO C to One

The “memory” hardware

A Table Of Contents

1 Data & Memory

- Basic datatype
- Arrays
- Structures
- Pointers
- “malloc()” …

2 Control Flow

- Syntax
- Conditionals: “if”, “switch”
- Loops: “for”, “while”

3 Library, Files & L. features

- “include” libraries
- rand() and RAND_MAX
- “fopen()” etc.
- #define

A Libraries

Each source file includes libraries independently.

You need some library, include it that source file.

A Files

How to open a file for read? For write?

Remember to close the file!

It's similar to IO with files and keyboard.

`fscanf(); scanf(); getchar(); fgetc(); printf(); fprintf();`

How to check if you have read in everything?

Check for the return value "End of file" **`EOF`**

A IO: Constructing string

Use `sprintf()` to construct string.

Think about the terminating character before you call them!

```
1   int main() {
2       char strOut[100] = "";
3       char *name = "Alan Turing";
4       int birth = 1912;
5       double possibility = 10e22;
6       sprintf(strOut, "%s, born in %d, built Bombe to "
7               "look for 1 in %f correct combination.",
8               name, birth, possibility);
9       puts(strOut);
10  }
```

A Random Numbers

What is `srand(time(NULL));`

And what header file to include if you need this?

What is `RAND_MAX`? What data type is it?

And what header file to include if you need this?

How to get a random number in a given interval

```
(double)rand() * ILength / RAND_MAX + lBound
```

A Strings `<string.h>`

Strings are character arrays that ends with `'\0'`

Think about the terminating character before you call them!

Comparing two strings

```
strcmp(char*, char*); strncmp(char*, char*,size_t);
```

Copying string from one place to another

```
strcpy(char* dest, char* src), strncpy();
```

Counting the length of a string: `strlen(char*)`

What is the `strlen("")` How many bytes is `""`

A Mathematical library

Trigonometry: `sin(x) cos(x) tan(x)`

Notice that the argument is in rad!

Common functions: `sqrt(x), pow(b, p)`

All arguments are in `double`.

Rounding: `ceil(x) floor(x) fix(x)`

Return type is still `double`, casting required!

A

Syntax: be careful with:

Semicolumns ;

No ";" after function def. Need ";" after `typedef`

Quotation marks "a" and 'a'

What is the datatype of "a" and 'a'?

`if (var == 0)`

Mind the number of "=". Better `if (0 == var)`

A

Syntax: general advice

Good coding style is important.

Make errors easy to find.

A

Syntax: Conditional statement

Common syntax with if-elseif-else

It's more clearer

```
1   int main(){
2       int cond1, cond2, cond3;
3       if (cond1) {
4           puts("Cond1 Holds");
5       } else if (cond2) {
6           puts("Cond2 Holds");
7       } else if (cond3) {
8           puts("Cond3 Holds");
9       } else {
10          puts("None Hold");
11      }
12      return 0;
13  }
```

A

Syntax: Conditional statement

`switch` statement, be careful with `break;`

And always comes with a default!

```
1   int main(){
2       int cond1, cond2, cond3;
3       switch (cond1) {
4           case 1: puts("case1"); break;
5           case 2: puts("case2"); break;
6           case 3: puts("case3"); break;
7           default:
8               puts("Other cases"); break;
9       }
10      return 0;
11  }
```

A Syntax: loops

What is wrong with the following code?

It's a common mistake.

```
1    int main(){
2        int cond1, cond2, cond3;
3        while (cond1);{
4            puts("I will not repeat");
5            puts("Guess why?");
6        }
7        return 0;
8    }
```

A

How to fix this code?

```
4   int main() {
5       printf("input the coefficients:");
6       double a,b,c;
7       scanf("%lf %lf %lf",a,b,c);
8       if (a==0&&b==0&&c!=0)printf("no root");
9       else if (a==0&&b==0&&c==0) printf("root are any real number");
10      else if (a==0&&b!=0)
11          double x=-c/b;
12      printf("%lf",x);
13      else if (a!==0){
14          double d=pow(b,2.0)-4.0*a*c;
15          if (d<0){printf("no real root");}
16          else if (d>=0) {double x1=((-b+sqrt(d)/(2.0*a);
17              double x2=(-b+sqrt(d)/(2.0*a)
18              printf("%lf/n %lf/n",x1,x2)};
19      }
20      return 0;
21  }
```

A

By first formatting it!

```
12      } else if (a == 0 && b != 0)
13          double x = -c / b;
14      printf("%lf", x);
15      else if (a ! = 0) {
16          double d = pow(b, 2.0) - 4.0 * a * c;
17          if (d < 0) { printf("no real root"); }
18          else if (d >= 0) {
19              double x1 = ((-b + sqrt(d) / (2.0 * a);
20              double x2 = (-b + sqrt(d) / (2.0 * a)
21              printf("%lf/n %lf/n", x1, x2)
22          };
23      }
24      return 0;
25  }
```

A

Syntax: general advice

Good coding style is important.

Make errors easy to find.

A

Datatype and Variables

Variable: Named bytes in memory

Type, size, range, scope, and address

A Type

`int, char`

Integers (And their variants):

`float, double`

Decimal numbers, (float points):

`char*, char[], int*, struct s*`

Pointer types, values are address

A Scope of variable

Global Var, DISALLOWED

Defined outside function

Local var, accessible within a function

Define in function. Vanish after the function exits

Returning address of local variable is always illegal

A Which code is wrong? Why?

```
1  #include <stdlib.h>
2  #include <time.h>
3
4  int *getRandomVector(int n) {
5      int a[100] = {0};
6      srand(time(NULL));
7      for (int i = 0; i < n; ++i) {
8          a[i] = rand();
9      }
10     return a;
11 }
12
13 int main() {
14     int* num = getRandomVector(10);
15     return 0;
16 }
```

```
1  #include <stdlib.h>
2  #include <time.h>
3
4  int *getRandomVector(int n) {
5      int* a = (int*)calloc(100, sizeof(int));
6      srand(time(NULL));
7      for (int i = 0; i < n; ++i) {
8          a[i] = rand();
9      }
10     return a;
11 }
12
13 int main() {
14     int* num = getRandomVector(10);
15     free(num);
16     return 0;
17 }
```

A User defined type: structures

typedef defines a type, not a variable!

Put it outside the functions! It's not g. var.

```
typedef struct {
    int member1;
    int memeber2;
    char* member3;
}uselessStruct;
```

A Pointers

Pointers means address

Just numbers. Passing pointer means passing address.

Pointer variable stores address.

So it “points” to a location in memory.
The type of it is the content the address contains

A Pointers

Reference operator: &

The result of this operator is an address.

De-reference operator: *

The result of this operator is the content in that address
Consider as this the “inverse” operator for “&”

A

Arrays (and pointers)

Continuous chunk of memory

The underlying reason why name of array is pointer

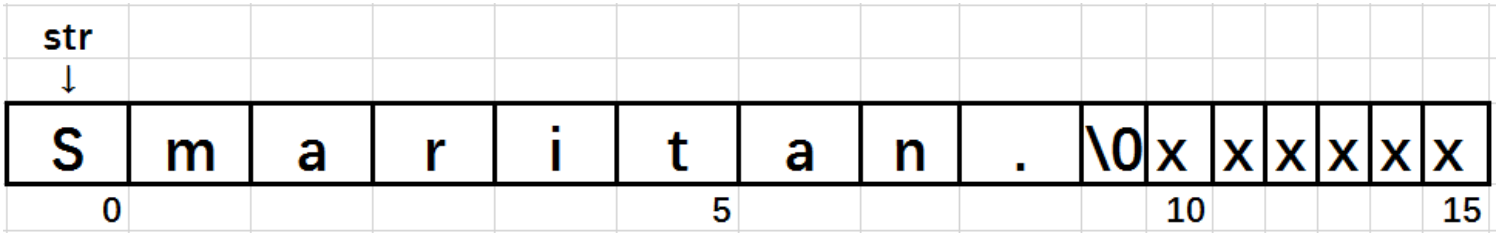
`a[i]` and `*(a + i)` is always equivalent

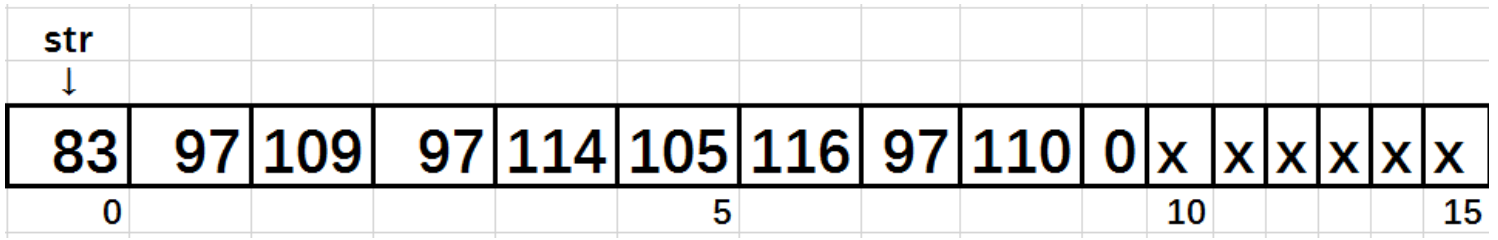
`a` can be the name of an array or an pointer
This is also the reason why array index starts with 0

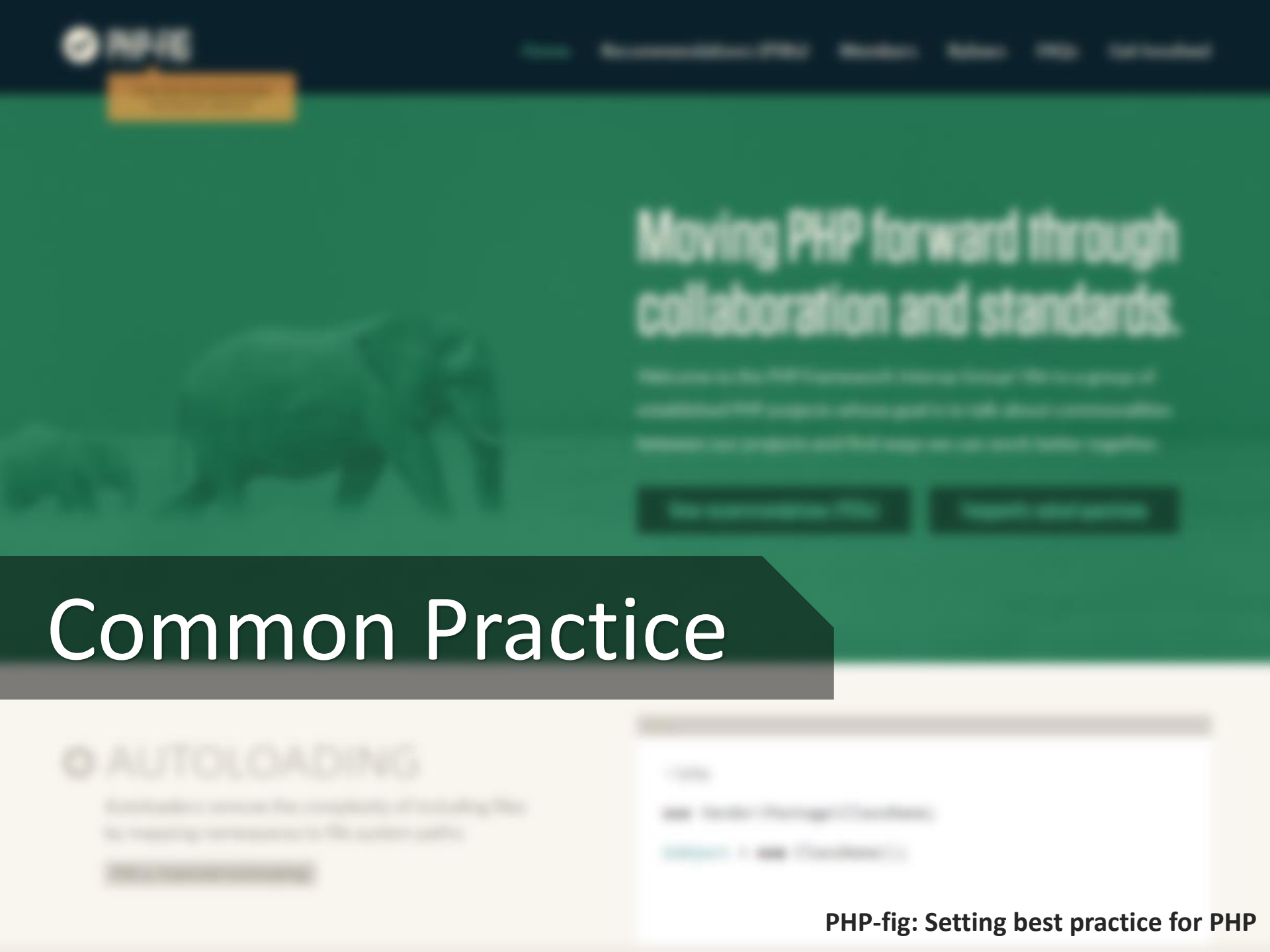
A Strings

`char str[15] = “Samaritan.”`

`str` is the name of an character array, a special array that each element is an ASCII character, and ends with **`\0`**







Common Practice

PHP-fig: Setting best practice for PHP

A

Tricks: How to deal with large "if"

NOT GOOD: "if" with a long condition

Hard to read/write, hard to change, confusing logic

Good: a function

Easy to copy/paste, easy to add/delete/change conditions

A

Bad practice

```
int main(){
    char a = 'a';
    if (a == 'a' || a == 'e' || a == 'i'
            || a == 'o' || a == 'u' || a == 'A'
            || a == 'E' || a == 'I' || a == 'O'
            || a == 'U') {
        puts("a is a vowel");
    } else {
        puts("a is consonant");
    }
}
```

A Good practice

```c
int isVowel(char c) {
    if (a == 'a' || a == 'A') return 1;
    if (a == 'e' || a == 'E') return 1;
    if (a == 'i' || a == 'I') return 1;
    if (a == 'o' || a == 'O') return 1;
    if (a == 'u' || a == 'U') return 1;
    return 0;
}

int main(){
    char a = 'a';
    if (isVowel(a)) {
        puts("a is a vowel");
    } else {
        puts("a is consonant");
    }
}
```

A Trick: dealing with const strings

NOT GOOD: Copy/paste string every time needed

Time consuming, very hard to change!

NOT GOOD: Use “switch” to select a string

Heavy coding, cost too much time

Good: Use a string array!

Easy to access. Easy to make changes.

A Good Practice

```
1   int main(){
2       char* dayNames[] = {
3               "Sunday", "Monday", "Tuesday",
4               "Wednesday", "Thursday", "Friday",
5               "Saturday"
6       };
7       puts("There are 7 days in a week");
8       for (int i = 0; i < 7; ++i) {
9           printf("%s, ", dayNames[i]);
10      }
11      return 0;
12  }
```

Person Of Interest (2015): “If-then-else”

A

In-class exercise:

Pointers: Read the code, and answer the questions

```
1   int* foo(int* p, int size) {
2       printf("%d", p[0]);
3       printf("%d", p[-1]);
4       printf("%d", *(++p));
5       printf("%d", ++(*p));
6       return p;
7   }
8
9   int main() {
10      int a[] = {1, 3, 5, 7, 9};
11      int *ptr = foo(a + 2, sizeof(a)/sizeof(a[0]));
12      printf("%d", *ptr);
13  }
```

A

In-class exercise:

Pointers:
Read the code, and answer the questions

Always: a[i] is equivalent to *(a + i)

Understand this using the memory point of view!

Adding / subtracting on a pointer is moving it around!

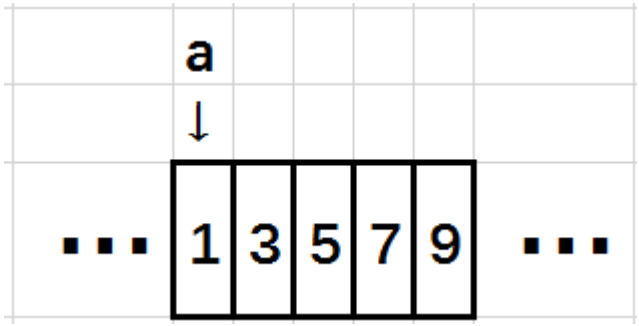
You are essentially adding/ subtracting the address

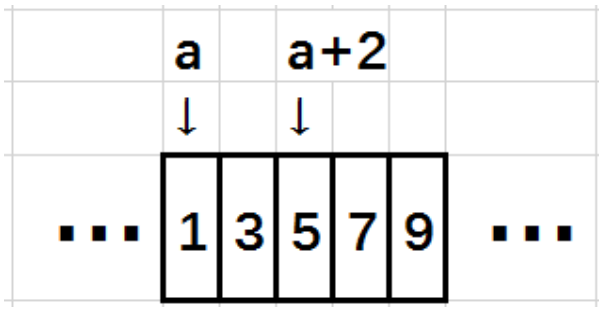
Passing an array to function == Passing a pointer

A

Pictures are your friends

```
 9  int main() {
10      int a[] = {1, 3, 5, 7, 9};
11      int *ptr = foo(a + 2, sizeof(a)/sizeof(a[0]));
12      printf("%d", *ptr);
13  }
```



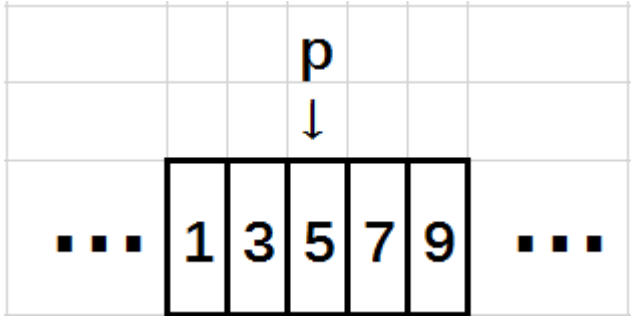


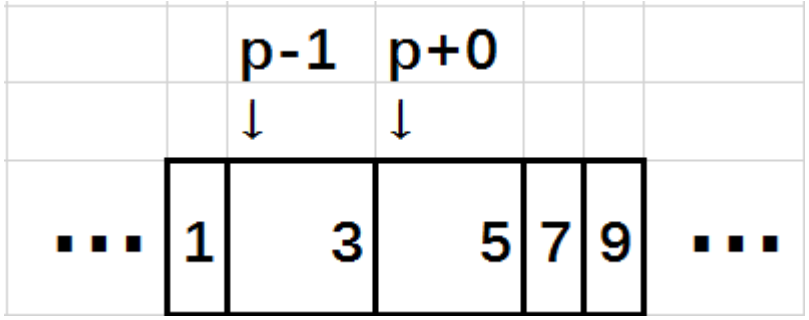
A

In-class exercise:

Pointers: Read the code, and answer the questions

```
1   int* foo(int* p, int size) {
2       printf("%d", p[0]);
3       printf("%d", p[-1]);
4       printf("%d", *(++p));
5       printf("%d", ++(*p));
6       return p;
7   }
```



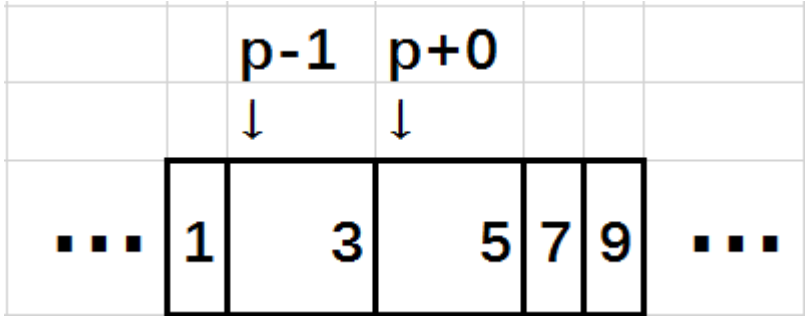


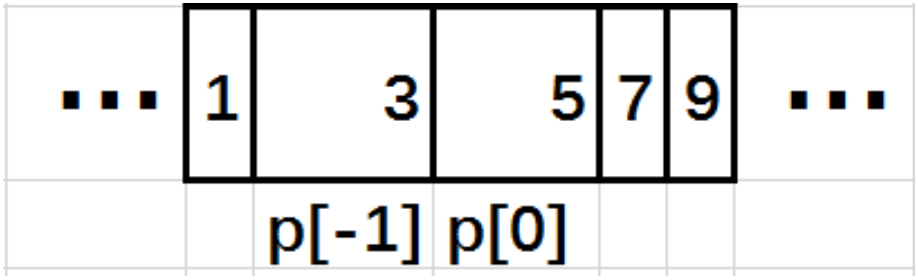
A

In-class exercise:

Pointers: Read the code, and answer the questions

```
1   int* foo(int* p, int size) {
2       printf("%d", p[0]);
3       printf("%d", p[-1]);
4       printf("%d", *(++p));
5       printf("%d", ++(*p));
6       return p;
7   }
```

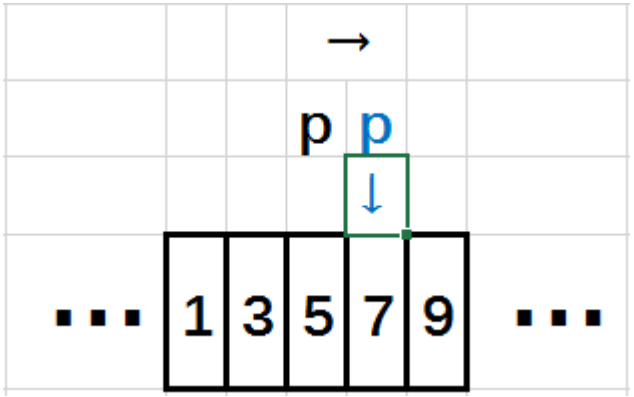


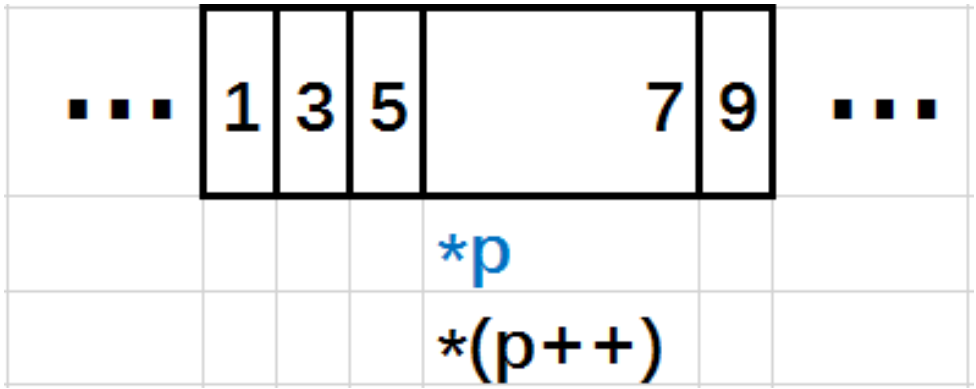


A In-class exercise:

Pointers: Read the code, and answer the questions

```
1   int* foo(int* p, int size) {
2       printf("%d", p[0]);
3       printf("%d", p[-1]);
4       printf("%d", *(++p));
5       printf("%d", ++(*p));
6       return p;
7   }
```



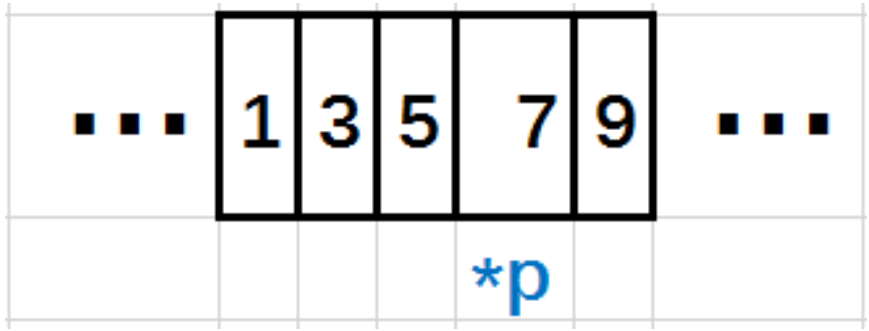


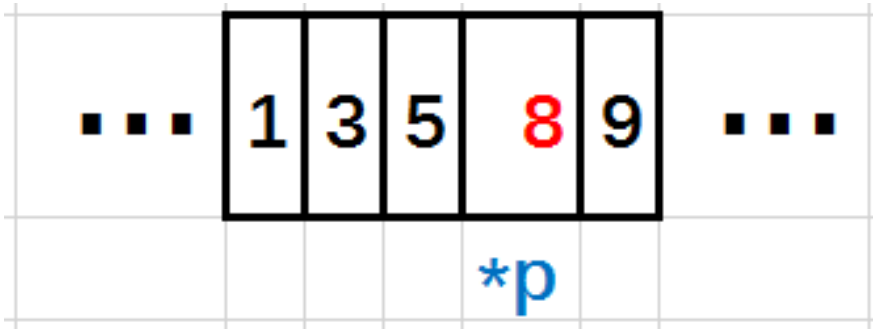
A

In-class exercise:

Pointers: Read the code, and answer the questions

```
1   int* foo(int* p, int size) {
2       printf("%d", p[0]);
3       printf("%d", p[-1]);
4       printf("%d", *(++p));
5       printf("%d", ++(*p));
6       return p;
7   }
```



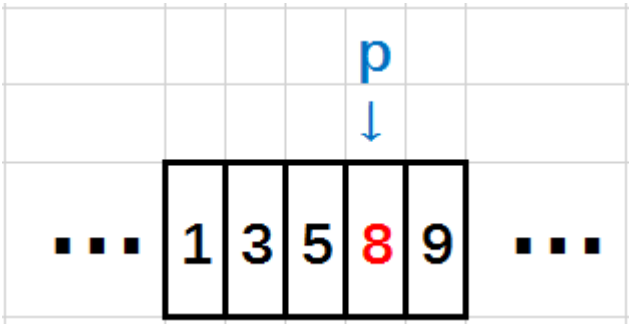


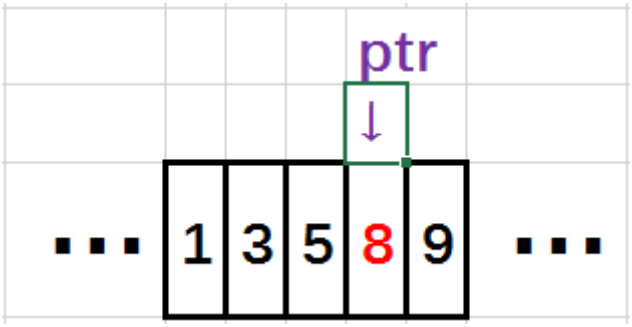
A

In-class exercise:

Pointers: Read the code, and answer the questions

```
 6          return p;
 7      }
 8
 9      int main() {
10          int a[] = {1, 3, 5, 7, 9};
11          int *ptr = foo(a + 2, sizeof(a)/sizeof(a[0]));
12          printf("%d", *ptr);
13      }
```





A

In-class exercise:

How to determine the size of a type?

What datatype is exactly one byte large?

How to determine the size of a structure type?

So you know how much memory to allocate for them

A

In-class exercise:

Find what is the ASCII code of character ‘;’ and ‘?’ without the Internet.

What is ASCII code? What’s the range?

What data type stores a character?

How to print it out?

What format should we use?

A

In-class exercise (HW6):

Design a function that reads unknown number of integers from file into an specified array.

How to design function argument?

Filename, destination array, size of destination array.

How to know whether we have read all numbers?

USE EOF!!

A

In-class exercise (h6e1):

1. **Design a structure to denote complex number.**
2. **Design a function for each of calculate the magnitude**

What consists a complex number?

Real part, imaginary part, perhaps more?

How to tell the function where to put the results?

Array/Pointer sytle, SIZE of array!

A Exam advices

ASK QUESTIONS!

It's your right to do so. Always ask!

Do not waste time on debugging!

Debug only small function. Do not spend too much time debugging

Start from the easiest! (IO, easy calc…)

Read all question, do the easiest ones first!

Common Questions

If not specified, how large is the input?

Typically less then 1k. My lucky number is 32767.

Is `char a[]` and `char* a` equivalent?

They are different. But functionality similar.

Have to use template? YES!!!

Responsibility is on you.

A Common Questions

How detail should we describe an algorithm?

See the lecture slides. “Pseudocode”

1. **Open fridge**
2. **Put the elephant in**
3. **Close the fridge**

NOT OK!

1. **Find a large fridge**
2. **While its too big:**
    1. **Keep looking**
    2. **Measure size**
3. **Put it in by ….**
4. **Close the fridge.**

OK

KEEP

CALM

AND

BEGIN

THINKING