

Анн-па.
Вонув, Уваров,
Капуст,
Основа Олександр.
ураган.
Тільки в енергії
беремо.

Сьогодні ж
до і уранку
по еху
с++;
Ванов

```
j = (*p++);
```

```

*s = *t;
if (*t != '\0') {
while (!isalpha)) {

```

```
++t;
```

```
*s = *t;
```

```
++s;
```

```

}
*s = '\0';

```

```
}
```

```

*s = *t;
if (*t != '\0') {

```

```
if (*t != '\0') {
```

```
while (!isalpha)) {
```

```
++t;
```

```
++s;
```

```

}

```

```
while ((*s++ = *t++));
```

~~while~~

```
while ((*s++ = *t++) && (*s != '\0'))
```

```
return *s;
```

```

(Parâmetro) (*s) - (Parâmetro) (*t);
(Parâmetro) (Parâmetro)

```

```

++s;
++t;

```

```

}
}

```

arg >= 1; arg < 0 = erro bur. parâmetro



fwrite
fstream

feof, ctrl + D 2 page → jump to
beg "1"

while ((c = fgetc(f)) != EOF)

mm um

```
do {
    while (*p == ' ') p++;
    while (*t != ' ') t++;
    if (
```

xterm
>

while(1)

> pwd - показать путь к кат. дир.

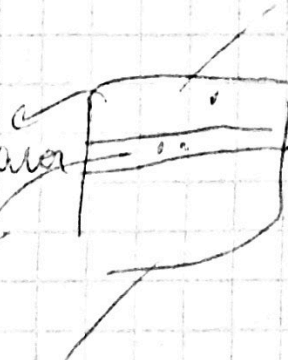
> cd .. - перейти в пред. директ.

> cd \$HOME - перейти к дир.

> ls - список файлов

> ls -al <имя дир.> - все файлы
им. вкл

им
дир. каталог
содерж
на пред
каталог



Содерж
дир.
кат. каталог

	имя файла	размер	дата	владелец
-	lwx lwx lwx			
d				
l				

Число файлов в кат.

> ~~big~~ first ^{1st} Second (1) Byger ogun gerupund
 > first - yg. uz 6 (runo m. ches. 7)
 (1) big first - new (yg. runo m. ches. 7)
 new -> first - 6 ~~gerupund~~ new.

> mkdir

> mkdir (runo m. ches. 7)

ln -s <new>; cp f1 f2 f1 f2

mv f1 d1
 cat > f1 - gerupund (2) - gerupund
 -> abc - 1

cat < f1
 no yman - 10
 Tail + n files
 Replne (new.) n espek gerupund.

grep substr files (> new file)
 my unipose meg espek (6 gerupund jamus)

scanf (argv[1], "%0", &x)
 unipose
 uz espek
 start...
 char *s = "xwxzwxzwx"

char *s = "xwxzwxzwx"
 for (int i=1, i<=9, ++i) { '

scanf, every %n;
~~scanf~~ while
 scanf("%[^\n]",
 fgets

~~char *s = "n";~~
 char *s = "2573";
 s[0] = '0';
 while s[0] != '\0'
 while (fgets(str, sz, fde))
 while (scanf("%n", of)
 Parameter must be %n

10000000.0 >>

↑
 zero. after no query.

Kn. pad.

int x = 15; strtol - convert to num
 k = sprintf(buf, MAX, "%d", x);
 char buf[MAX];
 c heap. memory
 copy = ...

Problems representation:

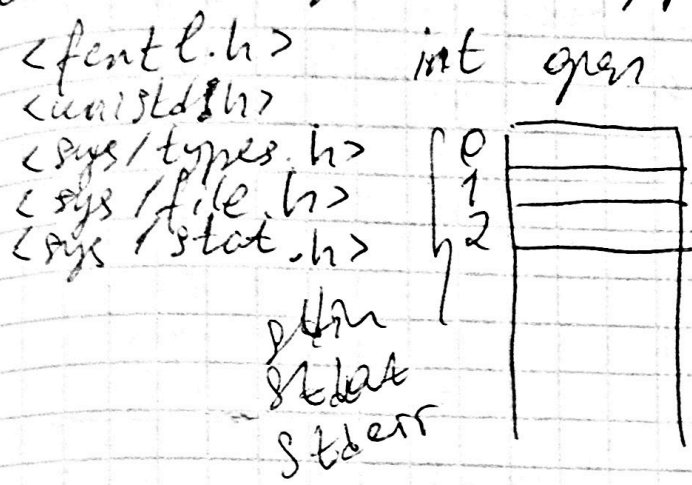
bool - builtin - sadd - overflow (int a, int b, int *res)

True - repen
 False - not

long long
 unsigned
 sadd
 sub
 sadd
 sub

uint32_t <stdint.h>

enum. borzobur - unrepresentable energy open. user. u. handle



Ung. group - unrepresentable

int open(const char *name, int flags, int mode)

fd = open("a.a", O_RDONLY);

enum. enum.

"w" = O_WRONLY |
 O_CREAT |
 O_TRUNC

≡ open("a.a", 0777)

- O_RDONLY
- O_CREAT
- O_WRONLY
- O_RDWR
- O_TRUNC
- O_APPEND
- O_EXCL

- lock
 - group
 - go to next

(, 0777) → 8 ~ 022 no group.

be vis. name for

the way to gain

fd = O_CREAT | O_APPEND | O_RDWR, 0777)

/
 write
 a much more

open(O_CREAT | O_WRONLY | O_EXCL) < 0, can gain

int close(int fd) // off_t seek(int fd, off_t int) mode
 < 0 error
 = 0 OK

0 = SEEK - SET - normal
 1 = SEEK - CUR - current
 2 = SEEK - END

size_t read(int fd, void *buf, size_t n)
 0, can grow
 < 0, error
 > 0, no-ops for some

write ()

fcntl f1, f2, 0750, fcntl
 mode no 512

fd1 = open(f1, RONLY, 0750);
~~fd2 = write(f2, ...)~~ fd2 = open(f2, O_CREAT | O_WRONLY | O_TRUNC, 0750);
 char buf[512];
 while (read(fd1, buf, 512) > 0) {
 int len = read(fd1, buf, 512);
 write(fd2, buf, len);
 }

FILE* → fd;

int fileno(FILE*);
fd → FILE*

FILE* fdopen(int fd, char* mode)
fclose(FILE*)

nanprunz "z+"

```
int zn, wn;  
char buf[1024];  
while (zn = read(fd1, buf, 1024)) {  
    char *p = buf;  
    while (zn) { wn = write(fd2, p, zn);  
                zn -= wn;  
                p += wn;  
    }
```

Read b
Jaagore
humans do
re repair.
write
read.

close);

~~for~~ int fd1 = ~~open~~open(f1, R O ONLY, 0777);

char buf[^{sizeof(double)}];
size_t size = lseek(fd1, SEEK_END, 0);
~~lseek(fd1, 0, SEEK_SET);~~

double
sizeof(double)

~~for (int i = 0; i < $\frac{size}{2}$; i++) {~~

~~char *p = buf; int wn;
 int zn = read(fd1, buf, sizeof(double));
 while (zn) { wn = write(fd2, p, zn);
 zn -= wn;
 p += wn; }~~

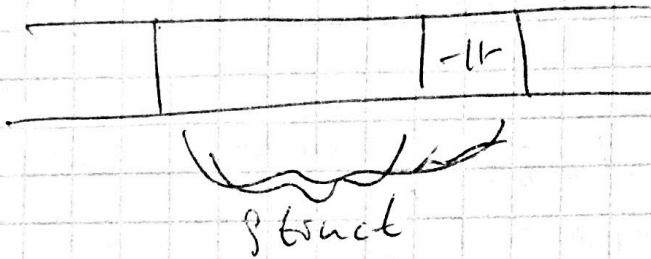
to page

struct S;

unsigned short size;

char data[512];

long long offset;



~~if (fopen(argv[2], "r")~~

~~if (fopen(argv[2], "r")~~

fd1 = open(argv[1], O_CREAT | O_RDWR);

seek(fd1, SEEK_SET, 0); ^{start} ~~seek(argv[2], SEEK_SET, 0);~~
 ; unsigned short count = 0; long long amount = 0; ~~char buf[8];~~
 while (count = seek(fd1, buf, 2)) { ^{char buf[8];}

long long offset;

~~sscanf(buf, "%us", &size);~~
 amount += size;

seek(fd1, SEEK_CUR, size);

read(fd1, buf, 8);

sscanf(buf, "%lld", &offset);

if (offset) {

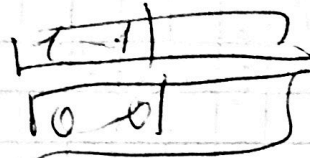
seek(fd1, SEEK_CUR, -size - 10); ^{offset}

printf("%lld", amount);


```
int fgetc(FILE *);
while ((c = fgetc(f)) != EOF)
```

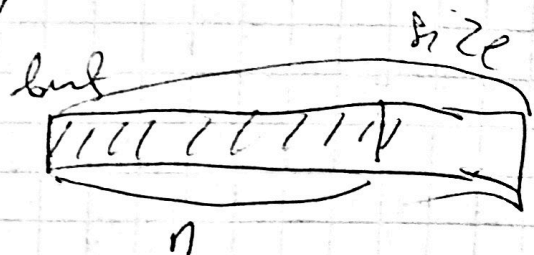
Print
EOF = -1
to int

signed char
unsigned char
int ← ✓



```
struct MyFile {
```

```
int fd;  
char *buf;  
int size;  
int pos;  
int n;
```



```
};
```

```
int Mygetc(struct MyFile *f) {
```

```
int fd; int count = 0;  
if ((fd = open("f", O_RDONLY, 0777)) == 0) {  
    return 1;
```

fgetc();

```
}  
if (pos < 0 || pos > n || buf[pos] == '\0') {  
    return EOF;  
if (pos < 0 || pos > n || buf[pos] == '\0') {  
    return EOF;
```

```
}  
if (count = read(fd, buf, size) < 0) {  
    return EOF;  
if (count = read(fd, buf, size) < 0) {  
    return EOF;
```

```
pos = 1;  
return buf[pos];
```

```
(unsigned char)
```


putc 'P'

void my_puts(int c, struct MyFile *f)

if ((n == size) || (buf[n-1] == '\n'))

write (

buf, f->pos++, 1, c;

union graina zepau. b propexu

<sys/stat.h>

struct stat {

<sys/types.h>

dev_t

st_dev;

name, uny. zepu.

dev_t

st_dev;

name, yep-la

link_t

st_nlink

name, nicsuue chf.

uid_t

st_uid

name, uny. zepu.

gid_t

st_gid

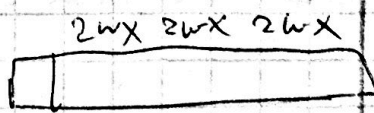
name, uny. zepu. k korpore

off_t

st_size;

mode_t

st_mode;



time_t

st_atime;

time_t

st_mtime

time_t

st_ctime; name, uny.

stat - сущ. сущ. сущ. $\rightarrow$ различия
предела
 ≥ 0 - OK
 < 0 - ошибка
 стат. к-т
 увелич. стат.

S - ISBuk

↓ gib-t gete gib (veit) - ~~was was was was~~
 ca. lang. br, um ja. 8 bit, was was
 um was to gib - gib cabrag.

W-OK 2

X-00 1

name wrong

```
int accessX(const char *name, int mode){
```

```
if (!getuid())
```

```
struct stat buf;
```

```
if (lstat(name, &buf) < 0){
```

```
return -1;
```

```
if (mode == 0) return 0;
```

```
if (buf.uid == getuid()){
```

```
if ((buf.st_mode >> 6) & mode == mode){
```

```
return 0;
```

```
} else return -1;
```

dirent.h>

```
DIR * opendir(const char *path);
```

```
struct dirent * readdir(DIR *);
```

```
while ((d = readdir(d)) != NULL)
```

```
    a_name | name | group | us stat;
```



Accessing & using the system.

```
while (dd = readdir(d))  
dd->d_name, ctime; stat
```

hangman, no game over

superint

```
if (lstat(path, &buf) < 0) continue;  
if (buf.st_mode & S_IFREG) {  
    ctime;
```

1. 1970-2.
2. 1970-2.
3. 1970-2.

time.h

```
#include <time.h>
```

```
time_t time(time_t *)
```

```
time_t t = time(NULL);
```

```
struct tm {
```

```
int tm_sec; [0-60]
```

```
int tm_min; [0-59]
```

```
int tm_hour; [0-23]
```

```
int tm_mday; [0-31]
```

```
int tm_year; [-1900]
```

```
int tm_wday; [0-6]
```

```
int tm_yday; [0-365]
```

```
int tm_isdst; [-1, 0, 1]
```

```
int tm_isdst; [-1, 0, 1]
```

1. 1970-2.
2. 1970-2.
3. 1970-2.

1. 1970-2.
2. 1970-2.
3. 1970-2.

1. 1970-2.
2. 1970-2.
3. 1970-2.

1 - memory.

struct tm * localtime (const time_t * t)
 returns, w/o exp. - local time (-11-), struct tm * tm
 (you to call. man.)
 time_t mktime (struct tm * tm); then we have
 same func.
 u tm isdst (-1)
 {anytime
 (high precision)

Levens

void f(int * year, int * mon, int * day, int offset)
~~time_t cur_t = mktime~~
~~struct tm t~~
 struct tm given_t;
 memset(&given_t, 0, sizeof(given_t));
 given_t.tm_mon = *mon - 1;
 given_t.tm_mday = *day;
 given_t.tm_isdst = -1;
 time_t t = mktime(&given_t);
 t += 3600 * 24 * offset;
 localtime(&t, &given_t);
 *year = given_t.tm_year + 1900;
 *mon = given_t.tm_mon + 1;
 *day = given_t.tm_mday;

Levens

l argum. kern. esp. Zager 2, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54, 55, 56, 57, 58, 59, 60, 61, 62, 63, 64, 65, 66, 67, 68, 69, 70, 71, 72, 73, 74, 75, 76, 77, 78, 79, 80, 81, 82, 83, 84, 85, 86, 87, 88, 89, 90, 91, 92, 93, 94, 95, 96, 97, 98, 99, 100, 101, 102, 103, 104, 105, 106, 107, 108, 109, 110, 111, 112, 113, 114, 115, 116, 117, 118, 119, 120, 121, 122, 123, 124, 125, 126, 127, 128, 129, 130, 131, 132, 133, 134, 135, 136, 137, 138, 139, 140, 141, 142, 143, 144, 145, 146, 147, 148, 149, 150, 151, 152, 153, 154, 155, 156, 157, 158, 159, 160, 161, 162, 163, 164, 165, 166, 167, 168, 169, 170, 171, 172, 173, 174, 175, 176, 177, 178, 179, 180, 181, 182, 183, 184, 185, 186, 187, 188, 189, 190, 191, 192, 193, 194, 195, 196, 197, 198, 199, 200, 201, 202, 203, 204, 205, 206, 207, 208, 209, 210, 211, 212, 213, 214, 215, 216, 217, 218, 219, 220, 221, 222, 223, 224, 225, 226, 227, 228, 229, 230, 231, 232, 233, 234, 235, 236, 237, 238, 239, 240, 241, 242, 243, 244, 245, 246, 247, 248, 249, 250, 251, 252, 253, 254, 255, 256, 257, 258, 259, 260, 261, 262, 263, 264, 265, 266, 267, 268, 269, 270, 271, 272, 273, 274, 275, 276, 277, 278, 279, 280, 281, 282, 283, 284, 285, 286, 287, 288, 289, 290, 291, 292, 293, 294, 295, 296, 297, 298, 299, 300, 301, 302, 303, 304, 305, 306, 307, 308, 309, 310, 311, 312, 313, 314, 315, 316, 317, 318, 319, 320, 321, 322, 323, 324, 325, 326, 327, 328, 329, 330, 331, 332, 333, 334, 335, 336, 337, 338, 339, 340, 341, 342, 343, 344, 345, 346, 347, 348, 349, 350, 351, 352, 353, 354, 355, 356, 357, 358, 359, 360, 361, 362, 363, 364, 365, 366, 367, 368, 369, 370, 371, 372, 373, 374, 375, 376, 377, 378, 379, 380, 381, 382, 383, 384, 385, 386, 387, 388, 389, 390, 391, 392, 393, 394, 395, 396, 397, 398, 399, 400, 401, 402, 403, 404, 405, 406, 407, 408, 409, 410, 411, 412, 413, 414, 415, 416, 417, 418, 419, 420, 421, 422, 423, 424, 425, 426, 427, 428, 429, 430, 431, 432, 433, 434, 435, 436, 437, 438, 439, 440, 441, 442, 443, 444, 445, 446, 447, 448, 449, 450, 451, 452, 453, 454, 455, 456, 457, 458, 459, 460, 461, 462, 463, 464, 465, 466, 467, 468, 469, 470, 471, 472, 473, 474, 475, 476, 477, 478, 479, 480, 481, 482, 483, 484, 485, 486, 487, 488, 489, 490, 491, 492, 493, 494, 495, 496, 497, 498, 499, 500, 501, 502, 503, 504, 505, 506, 507, 508, 509, 510, 511, 512, 513, 514, 515, 516, 517, 518, 519, 520, 521, 522, 523, 524, 525, 526, 527, 528, 529, 530, 531, 532, 533, 534, 535, 536, 537, 538, 539, 540, 541, 542, 543, 544, 545, 546, 547, 548, 549, 550, 551, 552, 553, 554, 555, 556, 557, 558, 559, 560, 561, 562, 563, 564, 565, 566, 567, 568, 569, 570, 571, 572, 573, 574, 575, 576, 577, 578, 579, 580, 581, 582, 583, 584, 585, 586, 587, 588, 589, 590, 591, 592, 593, 594, 595, 596, 597, 598, 599, 600, 601, 602, 603, 604, 605, 606, 607, 608, 609, 610, 611, 612, 613, 614, 615, 616, 617, 618, 619, 620, 621, 622, 623, 624, 625, 626, 627, 628, 629, 630, 631, 632, 633, 634, 635, 636, 637, 638, 639, 640, 641, 642, 643, 644, 645, 646, 647, 648, 649, 650, 651, 652, 653, 654, 655, 656, 657, 658, 659, 660, 661, 662, 663, 664, 665, 666, 667, 668, 669, 670, 671, 672, 673, 674, 675, 676, 677, 678, 679, 680, 681, 682, 683, 684, 685, 686, 687, 688, 689, 690, 691, 692, 693, 694, 695, 696, 697, 698, 699, 700, 701, 702, 703, 704, 705, 706, 707, 708, 709, 710, 711, 712, 713, 714, 715, 716, 717, 718, 719, 720, 721, 722, 723, 724, 725, 726, 727, 728, 729, 730, 731, 732, 733, 734, 735, 736, 737, 738, 739, 740, 741, 742, 743, 744, 745, 746, 747, 748, 749, 750, 751, 752, 753, 754, 755, 756, 757, 758, 759, 760, 761, 762, 763, 764, 765, 766, 767, 768, 769, 770, 771, 772, 773, 774, 775, 776, 777, 778, 779, 780, 781, 782, 783, 784, 785, 786, 787, 788, 789, 790, 791, 792, 793, 794, 795, 796, 797, 798, 799, 800, 801, 802, 803, 804, 805, 806, 807, 808, 809, 810, 811, 812, 813, 814, 815, 816, 817, 818, 819, 820, 821, 822, 823, 824, 825, 826, 827, 828, 829, 830, 831, 832, 833, 834, 835, 836, 837, 838, 839, 840, 841, 842, 843, 844, 845, 846, 847, 848, 849, 850, 851, 852, 853, 854, 855, 856, 857, 858, 859, 860, 861, 862, 863, 864, 865, 866, 867, 868, 869, 870, 871, 872, 873, 874, 875, 876, 877, 878, 879, 880, 881, 882, 883, 884, 885, 886, 887, 888, 889, 890, 891, 892, 893, 894, 895, 896, 897, 898, 899, 900, 901, 902, 903, 904, 905, 906, 907, 908, 909, 910, 911, 912, 913, 914, 915, 916, 917, 918, 919, 920, 921, 922, 923, 924, 925, 926, 927, 928, 929, 930, 931, 932, 933, 934, 935, 936, 937, 938, 939, 940, 941, 942, 943, 944, 945, 946, 947, 948, 949, 950, 951, 952, 953, 954, 955, 956, 957, 958, 959, 960, 961, 962, 963, 964, 965, 966, 967, 968, 969, 970, 971, 972, 973, 974, 975, 976, 977, 978, 979, 980, 981, 982, 983, 984, 985, 986, 987, 988, 989, 990, 991, 992, 993, 994, 995, 996, 997, 998, 999, 1000

struct tm cur_y; ~~not~~
memset(&cur_y, 0, sizeof(cur_y));
cur_y.tm_year = strtol(argv[1]) - 1900;
cur_y.tm_mon = strtol(argv[2]);
cur_y.tm_mday = -1; cur_y.tm_mon = 0;
mktime(&cur_y);

problem
cur_y
not to
on to newy.

l ignore for C int is
-1 => 1984

lpr dependent. m. delay

2. Zager write link (const char * path) can read
int randv (- 11) means random
int rename (const char * old, const char * new) (path can be
again. zip. - repeated. (zip. - repeated.
path. zip. - repeated.

closedir;

remove - yes. pattern

Sequence

uz 1 & 2, table re per. op., kagame
that's st-time bpen nokey. magerp
3. struct stat ~~stat~~
if (lstat(argv[1], &st) < 0) struct dirent * d;
DIR * dir = opendir(argv[1]);
while (d = readdir(dir)) {
printf(path, PATH_MAX, argv[1], d->dname);
if (lstat(path, &st) < 0) {
continue;

return 0;
char path[PATH_MAX];
while (d = readdir(dir)) {
printf(path, PATH_MAX, argv[1], d->dname);
if (lstat(path, &st) < 0) {
continue;

```

if (!ISREG(st.st_mode)) {
    if (st.st_mtime < 14 * 24 * 3600) {
        then - rename
        systemf("path-new, PATH=MAX, %s",
                argv[2], d -> d_name);
        rename(path, path-new);
        close(fd);
    }
}

```

seg. → 1-12 at. 14 seg. Characters per table.
 No. rows - 81.
 rows. 62.

```

int count = 0;
struct tm cur;
memset(&cur, 0, sizeof(cur));
cur.tm_year = strtol(argv[1]) - 1900;
cur.tm_mon = strtol(argv[2]) - 1;
cur.tm_isdst = -1;

int m = strtol(argv[3]) - 1;
while (cur.tm_mon == m) {
    if (cur.tm_wday >= 1 && <= 4)
        count += 8;
    else if (cur.tm_wday == 5)
        count += 6;
    cur.tm_mon++;
    cur.tm_isdst = 1;
}

```


if track dirent *

Chunup

```

argv[0] require type name
while (c = read(-1)) {
    for (int i = 1; i < argc; ++i) {
        struct stat st;
        if (stat(argv[i], &st) == 0) {
            if ((st.st_mode & S_IFMT) == S_IFDIR) {
                int var = 0;
                while ((c = read(dirfd(dir))) != -1) {
                    var++;
                }
                if (var == 2) {
                    close(dirfd(argv[i]));
                    printf(argv[i]);
                }
            }
        }
    }
}

```

Bag-2

```

argv[1], type & amount in max-amount-size-flag-seg.
between name & base in ~ path.
enum size > 0; between :
char st-size max-size = 0, st-size amount > 0;
char path[PATH_MAX];
struct dir *d = opendir(argv[1]);
struct dirent *de;
while ((de = readdir(d)) != NULL) {
    if (!strcmp(de->d_name, ".") || !strcmp(de->d_name, "..")) continue;
    if (sprintf(path, PATH_MAX, "%s/%s", argv[1], de->d_name) < 0) {
        continue;
    }
    if (ISREG(st->st_mode)) {
        amount += st->st_size;
        if (st->st_size > max-size) {
            printf("name: %s\n", argv[1]);
        }
    }
}

```



```
else if (strlen(st->name) > 4 & & strcmp
```

Задача 3

```
if (argc < 2, perror("Wrong number of arguments"),  
    if (fd = open("argv[1]", O_RDONLY) < 0) {  
        return 1;  
    }
```

4

see cur. bug. loop. 0 because of

because of wrong loop number. if you want to know.

```
int i, j, k, n = 1; w = 0;  
int i = 0; int ci = 0;  
struct tm t;  
for (int i = 1; i <= n; ++i) {  
    memset(&t, 0, sizeof(t));  
    t.tm_isdst = -1;  
    t.tm_mday = i;  
    mktime(&t);  
    if (t.tm_wday != 0 & t.tm_wday != 6) {  
        ++ci;  
    }
```

```
    if (ci == 4) {  
        iw++;  
        ci = 0;  
    }  
    if (iw == n) {  
        break;  
    }
```

if (iw == n) break;

Gcc

gcc -g prog.c -o prog

i
ptrace
core
q. parameter

gdb prog

.core

> help
> list (line #) > l; num => [num
> show listsize
> set listsize 50
> b(addr) - yes. r. address > b num > b addr
> delete #n, > d - nobr. yes. no step.
run <cpu. mem. exp>

Enter, nobr.
perme non.
kaman

> print <var>
> step
> next (the jst. b addr - 1)
> continue
> finish (barr. us q. or .)
breakpoint

> backtrace # full, parameter. for var. q.
> quit
3

gdb prog prog.exe

valgrind --leak-check=full prog

void rand(unsigned seed)

int rand(void) [0, RAND_MAX]
int MAX

[0, N)

$a = \text{rand}() / (\text{RAND_MAX} + 1)$
 $\text{rand}() / (N)$

[N1, N2] int $a = N_1 + (\text{rand}() / (\text{RAND_MAX} + 1)) * (N_2 - N_1 + 1)$
int

hand. gen. measure gamma were - flow

$[a_1, b_1], [a_2, b_2]$ parameters no unambiguous (orig. u⁺).

$a < a_1 \rightarrow a += a_1$
 else $a += a_2 - a_1$

f10 : f1.c ah

$\$(CC) \$(CFLAGS) -c \$ <$

gmake (Makefile)
 gmake -f <user>

$CC = gcc (=cc)$
 $CFLAGS = -Wall (-c)$

$\$(CC)$
 cc -c

$\1
 create base
 job - 2
 spooler
 run

all : res

res : f1.o f2.o f3.o

sec f1.o f2.o f3.o -o res
 f1.o : f1.c a.h
 gcc -c f1.c } f1.o : a.h

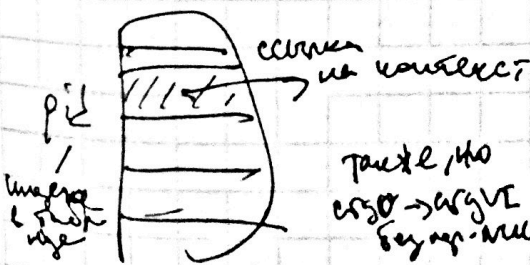
$\$(C -$
 uph job -
 uph 2
 $\$(W - 'res'$

clean : rm -f *.o

T. pozavala

New Services

name name



take, no
 error -> argve
 by up. null

exec (char *path, char *args, ... Null);
 execlp (char *file
 execl (char *system
 execv (char *path, ... Null, char *xenv)
 execlp
 execlp

Comment uog n gam. gann

part (c)

```
except ("ls", "ls", "-ol", 0);  
→ exit(1);  
0 x
```

```

uchar *mEO3 = { "ls", "al", NULL };
// except ("ls", epm);

```

Синтез глицерина и сложных
жиров кислот - ал!

getpid() - pid отъзв.

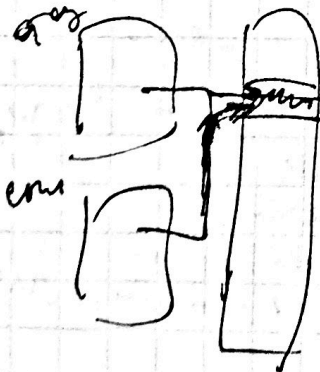
int wait(int *status); - ждет события. процесс
 не сам. выполняет (done, ...)

pid 20 - zartan. ogun uz chn. moyze

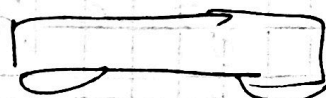
Если получится — спасибо тебе будет

Един в огуз не емане
брез-се процес замби

Наер-се: ~~БАН~~. откуп. градина



общий указатель



8
мозг.

8
сегмент.

взр. окончание


```

Ex 2. a.out p1 p2 ... pn sample range no
for (int i=1; i<argc; ++i)
    if (forks > 0) int p = fork(); // switch (fork)
    if (p < 0) {
        exit(1);
    }
    if (p == 0) {
        execlp(argv[i], argv[i], NULL); // exit(1);
    }
    else {
        wait(&status); // if (!WIFEXITED(status) || WEXITSTATUS(status) != 0)
    }
}
wait(&status); // - say no & course progress.

```

(WIFEXITED(status) && wait(&status);) // ! 0 means program terminated.
 !WEXITSTATUS(status) == 0) // yes/no

WIFSIGNALED(-1)
 WTERMSIG(-1)

Ex 3 n - zero never.
 n = atoi(argv[1], NULL, 0);
 for (int i=1; i<argc; ++i)
 printf("%d", argv[i]);
 --n;
 if (n == 0) {
 // ...
 }
 if (fork() > 0) // ...

```

int h = sizeof(argv[1], "d", &n);
int i=1; pid
while (i < argc) {
    pid = fork();
    if (pid < 0) {
        exit(1);
    }
    else if (!pid) {
        ++i;
        continue;
    }
    else {
        printf("%d", i);
        wait(&status);
        // ...
    }
}

```

for n. if (n == 0) fork

```

for printf("%d", i); 18
if (k == 1)
    pid = fork();
    if (pid > 0) {
        wait(&id);
        break;
    }
}

```

Задача 6. а. как ет. две gaining a.out gain
2 конс. require no sysbe

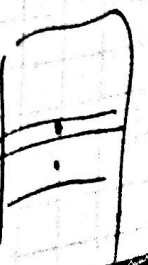
Уточно read / setbuf(f, NULL);

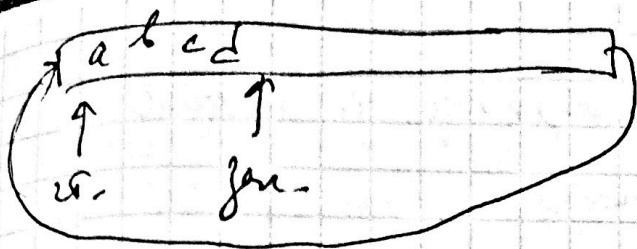
Комментар
~~(p1 || p2) && p3~~
int main(int argc, char *argv[]) {
 int call(char *name);
 int a;
 if (!fork()) {
 execlp(name, name, 0);
 _exit(0);
 }
 wait(&a);
 return (WIFEXITED(a) &&
 !WEXITSTATUS(a));
}

int main(int argc, char *argv[]) {
 return ((call(argv[1])
 || call(argv[2]) || call(argv[3]));
}

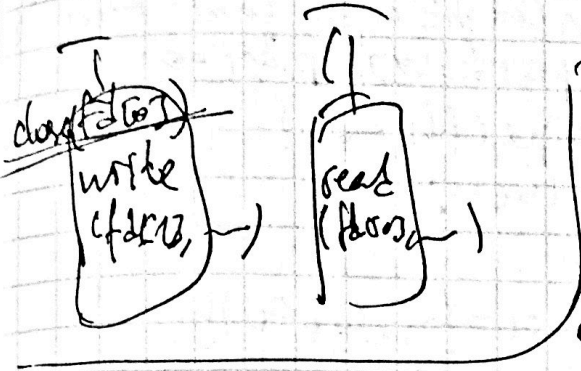
int pipe(int fd[2]), char ^{begin} → 0, no

just a - fd[0]
 just a - fd[1]

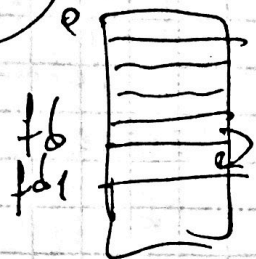




Чис. не должно быть - 0
 Конкр. чис. ато нег. жетер.
 !!! Все перемещать
 группировать



int dup(int old) - replace & drive
 int dup2(int old, int new) - Конкр.
 1. 2 → new
 fd = open(____, O_RDONLY);
 fd1 = dup(fd)



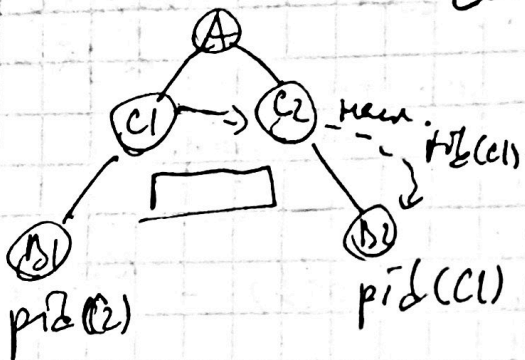
```


// ignore a.out pr1 pr2
int main(int argc, char *argv){
  if (argc < 2) return 1;
  int fd[2]; int pid1, pid2;
  if (pipe(fd) < 0){
    exit(1);
  }
  if (!pid1){
    dup2(fd[1], 1);
    close(fd[1]);
    execlp(p1, "close(fd[0]);",
    &close,
    if ((pid2 = fork()) < 0){
      exit(1);
    }
    else if (!pid2){
      dup2(fd[0], 0);
      close(fd[0]); close(fd[1]);
      execlp(p2, p2, 0);
      close(fd[0]);
      close(fd[1]);
    }
  }
}


```

Задача

Есть 2 процесса и массив



C1, C2 на массив.
на fork.

```

int main(int argc, char *argv)
{
    int pipefd[2]; pipefd[0] = 0;
    if (pipefd[1] < 0)
        exit(1);
    if ((pid_t) = fork() < 0)
        return 1;
}

```

4. 4. 4.

```

if (!pid_t)
    if (!fork())
        (close(fd[0]));

```

Кодовый

Пинг-панг
здесь 2-й процесс

```

int main(int argc,
        char *argv)
{
    int cur; int fd[2]; long N = strtoll(argv[1], NULL, 0);
    pipe(fd); pipe(fd+2);
    if (fork())
        close(fd[0]);
        close(fd[2]);
        while (read(fd[1], &cur, 1) == 1)
            print("%d", cur);
            1. 1. 1.
            if (cur > N) break;
            write(fd[2], &cur, 1);
    return 0;
}

```



анализируем... Таблица... последовательности... массивов.

```

close(fd[0]); close(fd[1]); while (write(fd[2], &cur, 1) == 1)
close(fd[2]); close(fd[2]);

```


1. Queue

```

int status, pld, fd[2], val = 0;
scanf("%d", &n);
pipe(fd);
if (!fork()) {
    close(fd[0]);
    close(fd[1]);
    write(fd[1], &val, sizeof(val));
}
while (wait(&status) != -1);

```

```

void
funf(int fr, int fw, int pid,
     int max) {
    int val;
    while (read(fr, &val, sizeof(val))
           == 4) {
        printf("%d\n", val);
        fflush(stdout);
        if (val > max) {
            close(fr); break;
        }
        close(fw);
        return 0;
    }
    write(____);
}

```

Ex 2 pr1 (prc 28 pr1) >> file - arquivo. na memória.
 eum ygame sac. => pr1

```

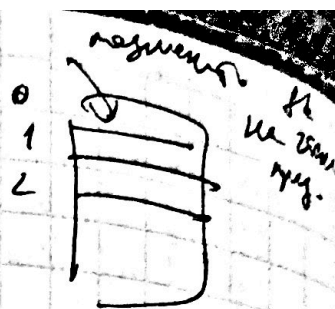
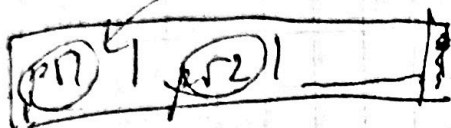
int main(int argc, char **argv) {
    int pld; int fd[2];
    if ((pld = fork()) > 0) {
        pipe(fd);
        if (fork()) {
            dup2(close(fd[0]));
            dup2(fd[1], 1);
            close(1);
            execvp(argv[1], argv, (char *) 0);
            exit(1);
        }
        close(fd[1]);
        dup2(fd[0], 0);
        int status;
        if (fork()) {
            execvp(argv[2], argv, (char *) 0);
            exit(1);
        }
        while (wait(&status) != -1);
        if (!status) {
            exit(1);
        }
    }
    while (wait(&status) != -1);
}

```

Uzun.

Zagora

p21 | p52 | — | p8a



```
int main (argc, char **argv)
for (int i=1; i<argc; ++i) {
    pipe (fd);
    if (fork()) {
        dup2 (f, fd[1]);
        execlp (argv[i], argv[i], (char *) 0);
        _exit(1);
    }
    dup2 (fd[0], 0);
}
if (fork()) {
    close (fd)
    execlp (argv[argc-1], argv[argc-1], 0);
}
```

New Seminar

*Specimen
discussed*

32 channels:
SYSTEM V

 $\langle \text{signal}, h \rangle$

4-138D

```
int kill(pid_t pid, int sig)
```

Wird (geteilt), —
Freie (SIGNI).

Энергичные сигналы - реакция на сигнал

СИБИРЬ некое государство (сразу убавает)

10-890p - кенже кезеңіне

SIG-TERM - tone gu-ue.

SIGINT - CTRL+C.

SIGCHILL - waste (—)

SIGALARM - alarm (cek.)

3164821

8164822

```
void (* signal(int sig, void (*handler)(int))) (int,
```

- возразить не
през. оград.
сущина

SIF - DFL
SL - LGN

Конец документа

Две вьюге сибирский клест, на вьюге белобрюх.

```

lagara { main()
        for(;;) pause 1;
    }

```

на 5-й стр. CTRL+C работает.

```

void handler(int signal) {
    signal(signal, handler);
    static int count = 1;
    if (count == 5) {
        SIGNAME(SIGINT, SIG_DFL);
    }
}

```

```

main
signal(SIGINT, handler);
for(;;) pause 1;
return 0;

```

в BSD kernel c опес. ; SYSTEM V kernel не

lagara ~~не~~ args броне, чем прогн бланк → exit(0)
 SIGALRM, бланк мора → на бланк.

```

void handler(SIGALRM) {
    printf("-1\n");
    fflush(stdout);
    _exit(0);
}

```

```

main (argc - time)

```

```

int sigaction(int sig, const struct sigaction *act,
              struct sigaction *old)
    (NULL)

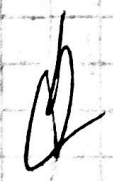
```

```

void (*sa_handler)(int);
int sa_flags; SA_RESTART не работает (броня. нех. нуль. нуль.
                потому не гана)
Пример sigaction(SIGINT, &struct sigaction) {
    .sa_handler = f,
    .sa_flags = SA_RESTART & NULL;
}

```

См. с. 111.




```
#include <unistd.h>
```

```
volatile int fl;
```

```
void main()
```

```
flag = 1
```

```
void do_work()
```

```
{ sigset_t ss, os
```

```
sigemptyset(&ss);
```

```
sigaddset(&ss, SIGUSR1);
```

```
while(1){
```

```
    while (!flag)
```

```
        sigsuspend(&os);
```

```
    flag = 0;
```

```
    sigprocmask(SIG_UNBLOCK, &ss, NULL);
```

```
}
```

Signal SIGUSR1, do-work

sigprocmask(0, NULL, &os);

<ipc.h>

<sem.h>

<shm.h>

<msg.h>

<types.h>

Key t fbox(char *frames, char c)

int semget(key_t key, int nsem, int flag)

argu

key to sem.

IPC_CREAT | 0666 (IPC_EXCL)

name arg.

argu - 1
prop. arg.

int semctl(int id, int num, int cmd, union semun arg)

GETVAL

{ int val;

SETVAL

ushort *array;

GETALL

struct semid_t *buf

SETALL

;

IPC_STAT

IPC_RMID

Oneprayer down:

while (!semctl(semid, 0, GETVAL, 0))

semctl(semid, 0, SETVAL, 0);

int semop(int id, struct sembuf *buf, int n);

{ short sem-num;

sem-op;

sem-flag;

he a. arg.

manage the

he genuine

arg.

can be sem. & buf

sem op

> 0

< 0, can flg=

> 0, can flg=

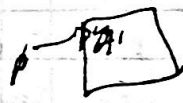
IPC_NOWAIT - *спрашивает*
 1 - ch., 0 - yes.
 struct sembuf down = {0, -1, 0}; semop(id, &down, 1);
 up = {0, 1, 0};

Задача

Нужно 0-ch.
 1-jan.

struct sembuf down[2] = {{0, 0, 1},
 {0, 1, 0}};
 up = {0, -1, 0};
 semop(id, down, 2);
 semop(id, &up, 1);

int shmat(key_t key, struct size, int flag);
 void *shmat(int id, void *addr, int flag);
 NULL 0
 align mem

char *p = shmat(id, 0, 0); *p = 'A' 

shmat(p); - *ошибка*

shmat(id, IPC_RMID, 0); - *быстро освобождает память*

Задача *очень сложная*

нужно 2 процессора, 2 ядра

int main(int argc, char **argv){
 char *p = shmat(atoi(argv[1]), NULL, 0);
 int id = semget(atoi(argv[2]), 2, IPC_CREAT | 0666);
 semctl(id, 0, SETVAL, 0); semctl(id, 1, SETVAL, 1);
 if (!fork()){
 while (fgets(str, 257, stdin) != 0){
 semop(id,

int main(int argc, char **argv){
 key_t my_key;
 scanf(argv[1], "%d", &my_key);
 int m_id, s_id;

for 2
 1-й процессор
 2-й процессор

на процессоре
 1-й процессор
 2-й процессор
 3-й процессор

100%
 100%
 100%

```

}
if (clockwork
    char *a;
    while (1) {
        if (strcmp(s_id, &down, 1) == 1)
        while (strcmp(s_id, &down, 1) == 0) {
            printf("%s", p);
            fflush(stdout);
            Sleep(s_id, &p, 1);
        }
        _exit(0);
    }
}

```

Shan Kp1;
Shan

$n=5 \Rightarrow$

7n. | page in $[1; 9] \times 0.1$ sec.
 24.10.17
 24.10.17 $[1; 9] \times 0.1$
 $S = \text{rand}() / (\text{RAND_MAX} + 1.0) \times 9 + 1$

memory
multiplex

usleep (500000)

or
multiplex

```
void h1(int s) {
    if (S == SIGINT) {
        semctl(s, id
```

```
int main(int argc, char **argv) {
    signal(SIGINT, h1);
    int count = atoi(argv[1], 0);
    int key = atoi(argv[2], 0, 0);
    int seed = atoi(argv[3], 0, 0);
    for (int i = 0; i < count; ++i) {
```

s-id = semset(key,
N,
IPC_CREAT
0666);

more forks

int.

```
for (int i = 0; i < count; ++i) {
    struct sembuf upgrade = {i, 1, 0};
    semop(sem, upgrade, 1);
```

```
for (int i = 0; i < count; ++i) {
    if (fork()) {
        srand(seed + i);
        double *talk, *cat;
        while (1) {
            talk =
```

```
if (semop(sem) == -1) { exit(0); }
```

```
}
signal(SIGINT, handler);
while (1) pause(1);
verb handler {
```


msg rcv c

turn 0 bec

for (int i=0; i<count; i++)
{ send(&msg, 1, SETA, 0); }

int msgget(key_t key, int flg)

struct msg {
long type;
};

msg;

int msgsnd(int id, void *buf, size_t n, int flg);

continue
case 1
sizeof(msg) - sizeof(long)
IPC_NOWAIT.

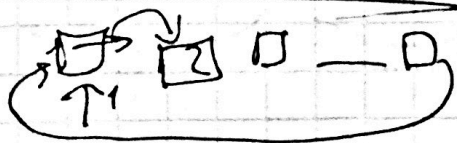
int msgrcv(-1, -1, -1, int flg)

long type

< 0 -> min(<abs> type)

msgctl(int id, IPC_RMID, 0);

case one 1



struct msg {
long type;
long val;
};

int main(-1) {
int n, max;
key_t key;
};

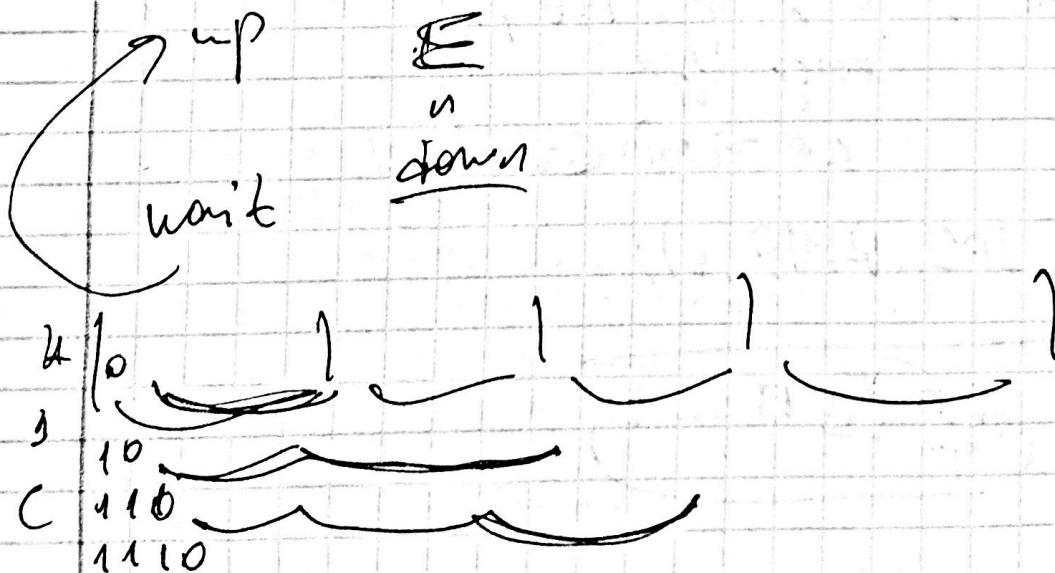
if (!fork())
while (1) {
msg - id = msgget(key, IPC_CREAT | 0666);
msg - id, long, sizeof(msg);
- sizeof(long) type, 0666

```

while(msg->msg-id, msg, sizeof(msg) -
      sizeof(msg_type), 0) == 0
{
    if (msg->val > max)
        msg->id, IPC_RMID, 0);
    printf("add %d", i, msg->val);
    msg->val++;
    msg->type = (i+1) % n + 1;
    msg->id, msg, sizeof(msg), 0);
    // size of msg
    exit(0);
}

```

Struktura msg msg = {



Землеустройство

Красная работа

Справочник, 1979; ANSI 1998 + STL

С++ - основана на абстрактном классе.

АТД - всё закрыто (private - доступным только изнутри)
классов инкапсулирует

или савиве
АТ сав сав
мисит.

Будет содержать
и значение
и указатель
на значение
наименование
полиморфизм.

использо-
вание функций

ген не нужен:

1) С++ - требует транслятора;

2) перепишет функции

перенос операторов

(структура - обертка)

такие параметры,

которые не используются

2) динамическое (вызв. ген.) - во время

3) параметры (модификаторы)

функции - транслятор генер.

```
#include <string.h> | #include <string>
                      | #include <iostream>
```

```
const char* str = "hello";
cout << str << endl;
```

```
char str[256];
cin >> str;
```

без префикса

вместо replace (std::)

getline(str, 256) - с преобразованием.

```
#include <iomanip>
```

```
cout << "hello" << endl;
```

// setprecision(3n), set, hex, dec, oct, base

```
namespace A { int f=1; }
```

```
namespace N { int f=1; int g=-2; }
namespace M { int f=1; int g=-2; }
```

```
int f=5; int main() { int f=1;
  cout << (N::f) << M::f << endl; }
```



```
using namespace N;
cout << f << g << endl;
```

```
using namespace M;
cout << f << M;
```

если. доп. переим.
указать в кодировке

Пр. 6. модуль именного пространства

```
namespace N { int f=1;
               int g=-2;
               void f(int, int) { ... }
               namespace M { int k;
                             void f(int) { f=5; }
               }
```

вызывает ошибку
life не вышло
using namespace
namespace не
вызывает ошибку
namespace

Then bool true, false

температура
не планет
или вычисления
(температура)

```
inline double sum(double a, double b) { return a+b; }
```

а) интерпретатор
- температура
б) монитор температуры
внешняя команда

```
struct St { int x;
```

static int y; — не будет использоваться в структурах

```
int St::y = 10;
```

```
cout << St::y & St obj; | St obj1, obj2; obj2.y = 20
      << obj.y
```

одно — только один
если есть структура
объект структуры
структура

```
int *p;
```

```
p = new int;
int(3) delete p;
```

p = new int(10); — массив, но используемый. удалить
delete [3]; сгруппировать конструкторы по умолчанию

Суммарная мемория наименования в базе аллокации $\Rightarrow p = (\text{nothrow}) \text{ new int } 10000;$

int a;
int &b = a; // b-адресу. mem.
// при изменении значения b не меняется a.

Пример: void f(long x) { x = 2; } // там & => 3 numb. -> 1 numb.

long a;
int &b = a
// => трансл. code: int temp = (int)a;
// int &b = temp;
// SS-const

Примеры: const char f(const long &x)

char & f(char &str) {
int i;

for (i=0; str[i] && str[i] != '\0'; i++);
return str[i];
}

(void f(const long &x))
=> ~~error~~.

main() { char str = "слова с пробел.";
f(str) = '\t';

Нельзя изменить const var: int & f() {
int x;
return x;
}

using namespace std;
// Пример float f(float &a) { a *= a; return a; }
// const into double
const double d = 1;
double d = 123.4;
cout << "0'k" << endl;
float z;
cout << (z = f(d)) << endl;

В class не гарантируется порядок.

Пример: class A { private:
int x;

public: void set(int a) { x = a; }
int get() { return x; }

// там также
// private =>
// абстракция
// данные
// не должны
// быть доступны

```
}; void A::set(int a){x=a;}
```

main() { Obj;

$$\text{obj.set}(101)$$

cont & obj.get();

1) this $\rightarrow$ x - newbawer ye. na
Bew. Bawer.

Конструкция выд.: две таб.: в начале при-н

Асint a) $\chi = a$, γ - констр. преобр-е

$$\Leftrightarrow A(\text{int } a) : x(a) \} \}$$

А obj2; - ошибка (по умолчанию не создается, если есть объект другой констр.)

Параметры по умолчанию. сдвиг в конце.

Элемент. транс. $\rightarrow$ 1 конструктор

Коротким.

5) $A_{obj} = A(10)$ - 2 конструктора

~~ex~~ ~~the~~ explicit - jumps nullosp. : obj 4 = 15;

```
class A {int x;
```

public: A(int a) { x=a; }

4.

main() { A *p1, *p2, *p3;

$$p_t = \text{new } A;$$

```

    p ← new A[LOW];

```

$p = \text{new } A(77);$

'delete p1;'

Delate 52p2j

Delete p3,

↓ размер выг. по размеру (по мет.)
⇒ ошукба

$\Rightarrow A(\text{cont } a=0) \{x=a\}$

Заг. мод. (6 момент змигує)
мод. (обробка числ. смт.)
перс

1. Визу и констр. абстракт.

Деление: "АЧЭЭ.
Всего 1

Иногда композитор сож. абст.
выс. же спаб. в конце пр.
нам вынес из себя выд.

```

class String { int len;
    char * str;
public:
    String(int len) { str = new char[len]; }
    String(char * s) { str = new char[strlen(s)+1];
        strcpy(str, s); }
    ~String() { delete[] str; }
}

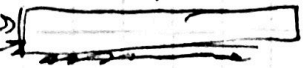
```

```

main() { String s1, s2(100), s3 = "сирокет";
        // 1000. 75.
        // (1 байт) '\0'
}

```

s2 = s3 (= - без пер. абр. (констр. абр. none))

s3.str → 

s2.str → 

String s4 = s2; (- констр. - р констр. вызов.)

1) по значению
2) по ссылке
3) по указателю

констр. с преем. вызовом 1 раз

```

String(const String & v) { len = v.len;
    str = new char[len];
    strcpy(str, v.str); }

```

C A z = A(2.5, u) - A(const A &x) { }

```

class I { int i;
public:
    I() { i(0); }
    I(int a) { i(a); }
    I(const I &b) { i(b.i); }
    ~I() { }
}

```

```

I f(const I &x, I y) { return y; }
main() { I i(0);
    I j(10);
}

```

s = s(1) // len
y = s(1) // len

f(s); const Ref
f(s); const Ref
f(s); const Ref
return y;

1234422f 14444

1.18

long long long long long Des Des 1 Des

Op. Overload

friend - доступ к записям на уровне

friend void f(A&);

Классы репрезентации # :: ?? . * typedef typedef

class C { double re, im;

создание
3 констр.

public: C(double a=0.0, double b=0.0) { re=a; im=b; }

bool operator==(C&a) { return re==a.re && im==a.im; }

operator== и unary ==

C operator+(C&a) { C tmp; tmp.re=re+a.re;

tmp.im+=a.im+im; return tmp; }

→ // & return C(re+a.re, im+a.im); }

friend C operator+(const C&a, const C&b) {

return C(a.re+b.re, a.im+b.im); }

-- → C operator-(C&a) { return C(-a.re, -a.im); }

++ → C& operator++(C&a) { a.re++; a.im++; return *this; }

++ → C& operator++(int) { C tmp(re, im); re++; im++;

(++a) → a.re++ a.im++

С++ + констр на уровне

применения

cout << "1" << endl; → friend ostream & operator<< (ostream& os, C&a) {

os << "re=" << a.re << "im=" << a.im << endl; return os; }

Операторы & = []() можно переопределить. Только внутри

классов (как метод)

Но у нас - то констр.! констр. не у нас

жесть.

К. констр.

прив. констр.

C Operator = (C & a) { re = a.re; im = a.im; return *this; }
 can't give pow with x3 = -x1; some expression.

operator double() { return ~~the~~ *this; }
 main() { double x1 = 25.5, x2;
 { a(1, 2), b(2, 1);
 { c = a + x1; } - recognizing number

↓
 double
 c = 10 + 6;
 { new double with c's friend → recognizing.

<pre> class String { char *str; int len; public: char operator[](int i) { return str[i]; } String operator+(String &s) { return String; String tmp; tmp.len = this->len + s.len; tmp.str = new char[tmp.len]; strcpy(tmp.str, str); strcat(tmp.str, s.str); return tmp; } String &operator=(const String &s) { delete[] str; (if (str == s.str) return *this; len = s.len; str = new char[len]; strcpy(str, s.str);) } </pre>	<pre> String c1("abcd"), c2("klm"), c3; c1[1] = 'A'; c3 = c1 + c2; </pre>
---	---

```

struct Prod {
  bool operator()(int v) { return v < 10; }
  void f(int *mas, int n, Prod c) { }
}

```

Задача 5.1.10

<u>1 17;</u>	sun
<u>17 120;</u>	venus 60
<u>120 400</u>	venus 400 moon
<u>f(1, 12);</u>	earth 420 venus 1000 moon moon

Кл. работа

Передача функции (стат. полиморф.)

Точнее
собнар.
типов.

$T \leftrightarrow T$
 $T \leftrightarrow T^*$
type def

- 1) по кат - без пар.
- 2) точнее кон - тинд

одно параметр

Вариант
реши

2)

float $\rightarrow$ double
bool $\rightarrow$ int
enum $\rightarrow$ enum
char $\rightarrow$ char
signed char $\rightarrow$ int
wchar $\rightarrow$ int
short $\rightarrow$ int
unsigned short $\rightarrow$ int

enum

enum $\rightarrow$ enum

double $\rightarrow$ long double
float $\rightarrow$ double

Станд.
меход.

3)

enum $\rightarrow$ enum
void $\rightarrow$ void * (name const $\rightarrow$ const void *)
 $\rightarrow$ enum + enum.

Вариант
меход.

4)

enum $\rightarrow$ enum
A(int x)
operator < > (int)

5)

(--) - разн-е кон-то параметров

Задача:

- 1) void f(int x)
- 2) void f(int)
- 3) void f(char)
- 4) void f(long)
- 5) void f(unsigned char)

одно параметр

несколько параметров

Станд. Те что не подходит

- 1) по кат-бу
- 2) с генерацией (enum)
- 3) где каждый параметр. код. мн-во ср. уг-ик 2/1
- 4) реверс

void f(int);
void f(double, int);
f(17) - некор

Задача

enum $\rightarrow$ enum
{ 1, 2, 3, 4 } - 1, 2, 3, 4

enum & enum
enum $\rightarrow$ enum
enum $\rightarrow$ enum

Class A { int b;
A obj;
public:

$B(int x, \dots) \rightarrow obj(x), \dots$

resources are always located in memory

class X { public: int static int i; } static static for static

```
int x::i = 15;
int main() {
    cout << x::i;
    x::print();
}
```

можно использовать в классе
можно использовать
в стат. области.

```
class A { static int a;
        int b;
    public:
        int A::a;
        int A::a;
        main()
        { const A a; }
}
```

- глоб. переменная по
умолчанию.

void f() { const int i; } не изменяемая переменная
является статической и ~~не~~ mutable

1) статическая переменная в классе не является
объектом. Статическая переменная
является объектом. // не является объектом
или виртуальной.

const м.б. статическая константа или ~~статическая~~

оп. f const int. Объявление

```
int main() {
    const A a;
    A x = 3;
    cout << a.y << a.x << a.f();
    << endl;
}
```


27/11/17

```
template <class T>
T abs(T&x) { return x>0?x:-x; }
```

abs<long>(a)

Еще есть вариант оп. с табулом все равно
нап. → long. вариант, а не математика

В стандарте есть универсальный: void, float, double, long, int
 template <class T, int SIZE=10>

```
class Stack { T st[SIZES];
int top = 0;
public: void push(T x);
void pop(); }
```

```
template <class T, int SIZE>
void Stack::push(T x) { if (top+1 > SIZE)
exit(10);
else { ++top; st[top] = x; }
```

```
template < >
T Stack<T, SIZE>::pop() { }
```

STL

- 1) контейнер
- 2) итератор
- 3) алгоритм

map (key, value)
 multimap (key)
 set
 multiset

Use: 1) input →
 2) output →
 3) forward →
 4) bidirectional →
 5) Random-access

++, --, !, =, ==, <, <=, >, >=, !=, <!, >!, <=!, >=!

Problem: how many unique ops:

- 1) iterator
- 2) reverse_iterator
- 3) const_iterator
- 4) const_reverse_iterator

```
vector<int> v1;
        v2(10);
        v3(10, 20);
int nums = {1, 2, 3, 4};
        v4(nums, nums+4);
```

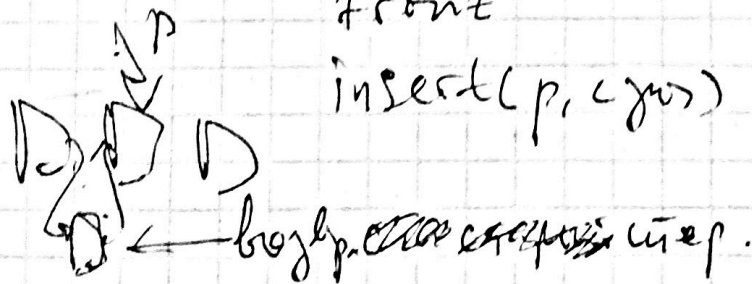
oops no thing up. rather only. This is the

reverse-iter. ++ and -- op. ←

Py-um: size(), capacity(), empty(), push-back (e.g.)
pop-back()
back()

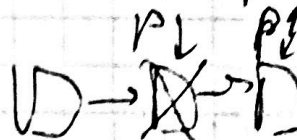
iterator, value-type

One more: push-front
pop-front
front

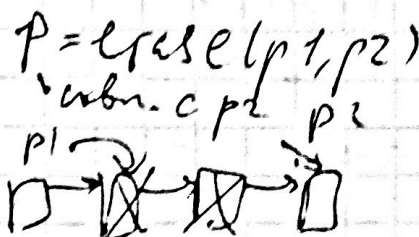


insert(p, <num>)

pt = erase(p)



Example: `vector<int> v;`
`vector<int>::iterator p = v.begin();`
~~while 1~~
`p = v.end();`
`while (p != v.begin()) { cout << *--p << ' ';`




```
String(const String &v) { len = v.len;
                          str = new char[len];
                          strcpy(str, v.str);
```

```
String full String temp;
return temp; }
```

```
x = f();
```

```
String (String &&v) { len = v.len;
                    str = v.str;
                    v.str = NULL; }
```

! конструктор
репрезентации

```
• nullptr;
  nullptr -> nullptr | void f(char *);
                    void f(int);
                    f(NULL); f(nullptr)
```

```
class X { }; X *p = new X; X *p2 = new (p) X;
```

```
void* operator new (size_t size); - специф. для
void* operator delete (void*) - специф. delete p2;
X *p = (X*) operator new (sizeof(X));
static_cast<X*> (—) Копирование
```

```
X *p2 = new (p) X;
p2 -> X();
operator delete (p);
```

```
class ComplexStack { Complex *p2;
                    sizeof - used, n;
```

Констр. конструктор: ComplexStack (int comp - &v)

used = v.used

n = v.n;

pt2 = static_cast<complex*> (ii operator new (n * sizeof (pt2[0]))

for (int i = 0; i < used; i++)

new (&pt2[i]) complex (v.pt2[i]);

complex_stack() { for (int i = 0; i < used; i++)

pt2[i].~complex();

← :: operator delete (pt2); }

template <typename T> void

f(T& cont) { T::iterator it = T.begin();

typename while (it != T.end()) {

it = insert(it, *it);

it += 2;

}

Value type

template <class T> void

f(T& t) { typename T::iterator it = T.begin();

while (it != T.end()) {

sum += *it;

it += 2; it++;

return sum;

}

}

class A { int a;

public: void f() { cout << a; }

class B: public A { int b;

public: void f() { cout << a << b; }

main() {

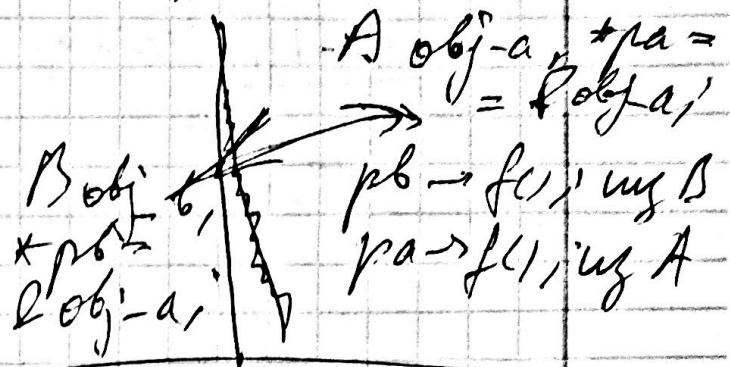
A obj-a, xpa;

B obj-b, xpb;

private none A ke byun B se onyabacum use
kon protected / private ke krunge bas.

конец оставшегося токена не
 protected: $\Rightarrow$ protected, public A $\rightarrow$ private B

при переп. функции:
 при-и hang-и в опис-
 и-и не одинаки вычислят
 и —



$$pb = (B^*) pa;$$

при записи:

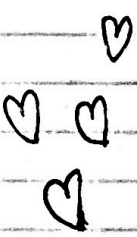
$$pa = (A^*) pb$$

$$pb = (B^*) pa;$$



не
 valac
 type

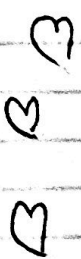
Функция с опис. именем
 в послед. конце
 закр. вызова,
 $f() \parallel E_2$
 $f()$ по
 $f(5, 1) \parallel E_2$
 $X = 2;$



при вызове

Сначала констр. вызова
 перед прог-зе

Действие в
 объектах пер.



Sp 154.2 ~~bonpac 3 x 1000~~ p-t(5) ?

154.5 p-y; ^{noisy boy. sayd. u.} ^{u. noy. x} ^{the} ^{cost}

154.16

~~#~~ fcntl(fdes, F_SETFL & ~O_NONBLOCK);

FILE *fr;

fr = fdopen(fdes, "r");

fscanf(fr, "%d", &u);

fprintf(fr, "%d\n", u); fgets
fflush(fr);

fclose(fr);

open(, O_RDONLY, 0666)

open(, O_WRONLY | O_APPEND | O_CREAT, 0666)

~~- библиотека -~~

u1

			(((
A	0 31 72		741
B	10		
C	110	3	29
D	1110		
E	1111		