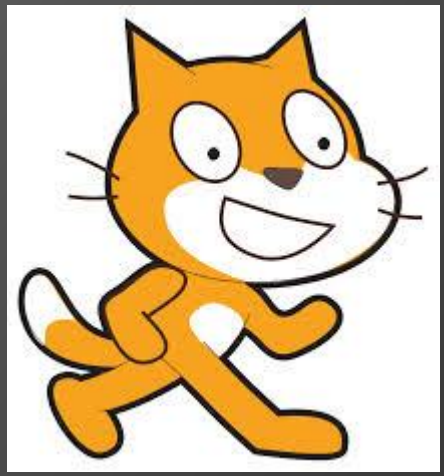
SCRATCH*



Introdução à lógica e programação

Matheus Schiavini
PIBID FÍSICA 2014
UNIPAMPA CAMPUS BAGÉ

*O presente trabalho foi realizado com apoio do Programa Institucional de Bolsa de Iniciação à Docência – PIBID (Edital 2013), da CAPES - Coordenação de Aperfeiçoamento de Pessoal de Nível Superior-Brasil.

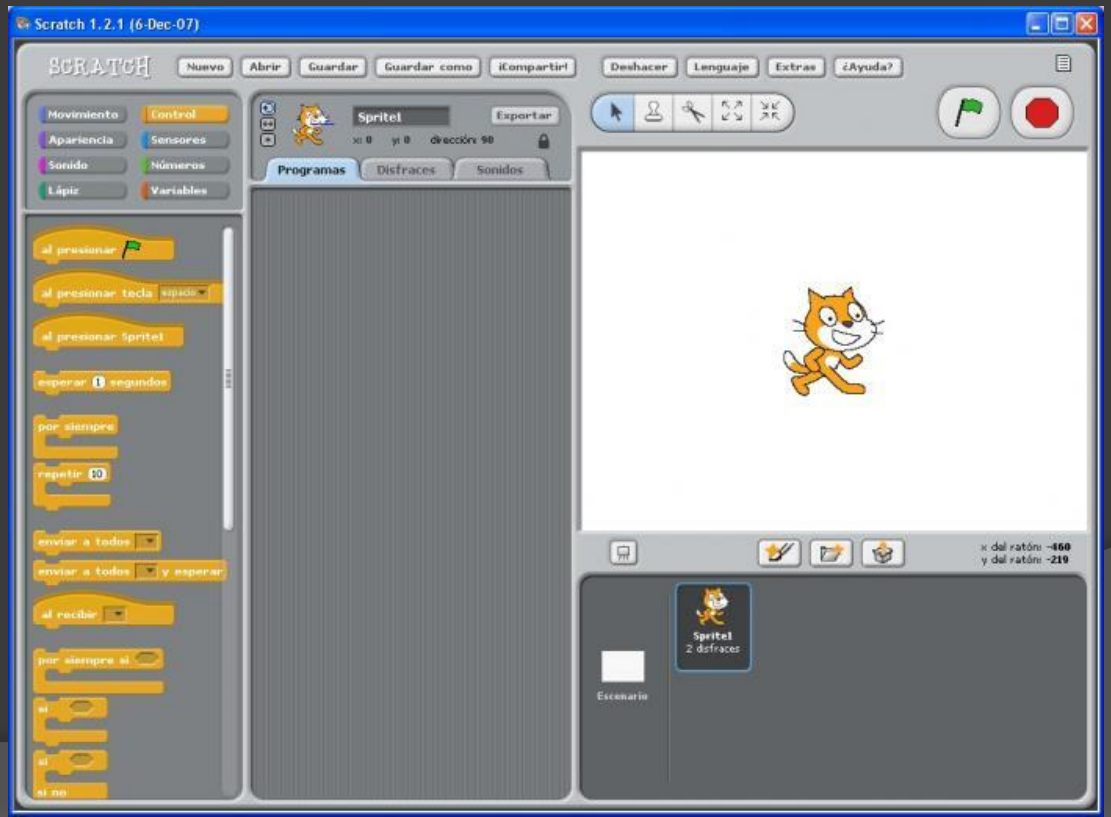
Atividades

- Instalação do SCRATCH
- Introdução à lógica de Programação
- Atividades práticas

O que é o Scratch?

É uma plataforma com fins didáticos, para introduzir a programação de jeito fácil e rápido a qualquer classificação de usuário.

Ele envolve a programação em bloco com um design divertido e simples.



É necessário o uso de:

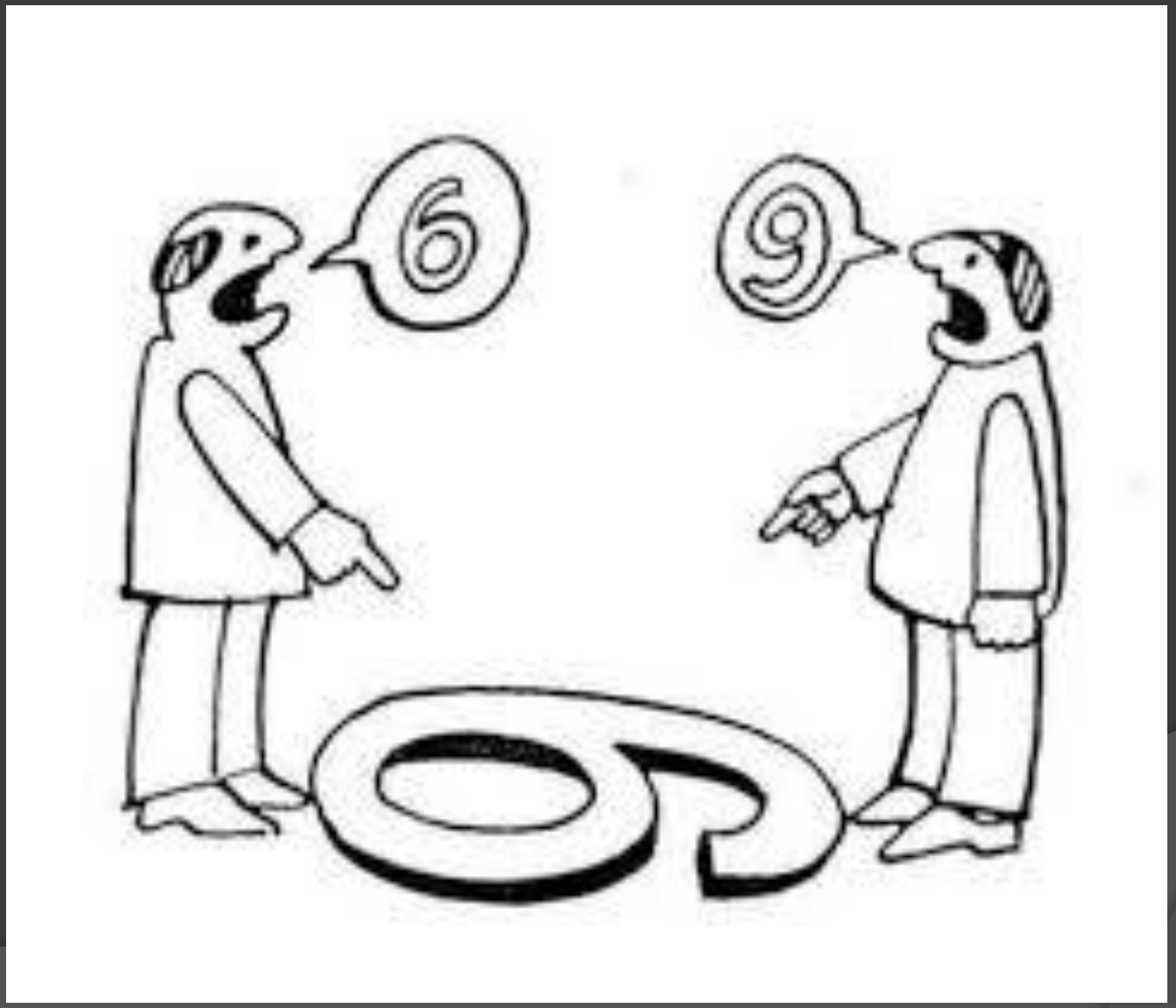
- Lógica de programação;
- Operadores lógicos;
- Ferramentas.

Lógica

Para programarmos em “scratch” precisamos primeiro entender o que é a lógica de programação.

É a técnica de desenvolver algoritmos (sequências lógicas) para atingir determinados objetivos dentro de certas regras baseadas na Lógica matemática e em outras teorias básicas.

Cada pessoa tem uma lógica diferente para usar!



Operadores lógicos:

- AND (E);
- NOT (NÃO);
- OR (OU);

- Obs: há também os menos usados como: NOR, XOR, XNOR e NAND;

Operadores lógicos trabalham com um tipo de informação, que deve entrar no sistema e ser interpretado pelo operador lógico.

Informações que entram no sistema podem ser chamadas de variáveis, que vão ser analisadas de acordo com o operador ou processo.

AND:

- Operador lógico no qual a resposta da operação é verdade (1) se ambas as variáveis de entrada forem verdade:

NOT:

- O operador “do contra”, ele inverte a variável de entrada e é visto de forma mais simples:

OR:

- Operador lógico no qual a resposta da operação é verdade (1) se pelo menos uma das variáveis de entrada for verdade:

Ferramentas:

Algumas outras ferramentas que são achadas para programar com o scratch, são os operadores numéricos (‘+’; ‘-‘; ‘=’; ‘*’; ‘/’;), comandos de movimento (andar XX passos; apontar para direção XX;), e os laços de repetição.

* Laços de repetição:

- Laço de repetição tem por função repetir alguma função de várias formas:

-> SEMPRE: a função dentro do laço será repetida para sempre;

-> SEMPRE SE: só ira repetir/acontecer caso haja uma condição a ser satisfeita;

-> REPETIR ATÉ:

Um número XX de vezes que pode ser pré-determinado ou informado;

Até satisfazer a função;

Até não satisfazer mais a função;

Referências

Scratch. Disponível em: < http://scratch.mit.edu/projects/editor/?tip_bar=getStarted > Acesso em: 06 novembro de 2010.

UNIVATES. CENTRO UNIVERSITÁRIO. Manual do Scratch. In: II OLINFU, 2010.

FORBELLONE, A. L. V., EBERSPACHER, H. F. Lógica de programação : aconstrução de algoritmos e estruturas de dados. São Paulo : Makron, 1993.

NAND:

- Operador lógico que é oposto ao AND;

A B s

0 0 1 -> todas as entradas falsas -> saída verdadeira

1 0 1 -> uma entrada verdadeira e outra falsa -> saída verdadeira

0 1 1 -> uma entrada falsa e outra verdadeira -> saída verdadeira

1 1 0 -> ambas entradas verdadeiras -> saída falsa

NOR:

- É o operador oposto do OR, ou seja a saída será verdadeira se as duas entradas forem falsas:

  A B s

  1 1 0 -> todas as entradas verdadeiras -> saída falsa

  1 0 0 ---> uma entrada verdadeira e outra falsa -> saída falsa

  0 1 0 ---> uma entrada falsa e outra verdadeira -> saída falsa

  0 0 1 ---> ambas entradas falsas -> saída verdadeira

XOR:

- O operador XOR, seria o exclusivo, ou seja, a saída será verdadeira se e somente se UMA das entradas forem verdadeiras, logo se tivermos duas entradas verdadeiras a saída será falsa:

  A B s

  1 1 0 -> todas as entradas verdadeiras -> saída falsa

  1 0 1 -> uma entrada verdadeira e outra falsa -> saída verdadeira

  0 1 1 -> uma entrada falsa e outra verdadeira -> saída verdadeira

  0 0 0 ---> ambas entradas falsas -> saída falsa

XNOR:

- É o oposto ao XOR, como no XOR tínhamos saída verdadeira quando somente UMA das entradas forem verdadeiras, com o XNOR a saída será verdadeira se as entradas forem iguais:

  A B s

  1 1 1 -> todas as entradas verdadeiras -> saída verdadeira

  1 0 0 -> uma entrada verdadeira e outra falsa -> saída falsa

  0 1 0 -> uma entrada falsa e outra verdadeira -> saída falsa

  0 0 1 -> ambas entradas falsas -> saída verdadeira

Obrigado!

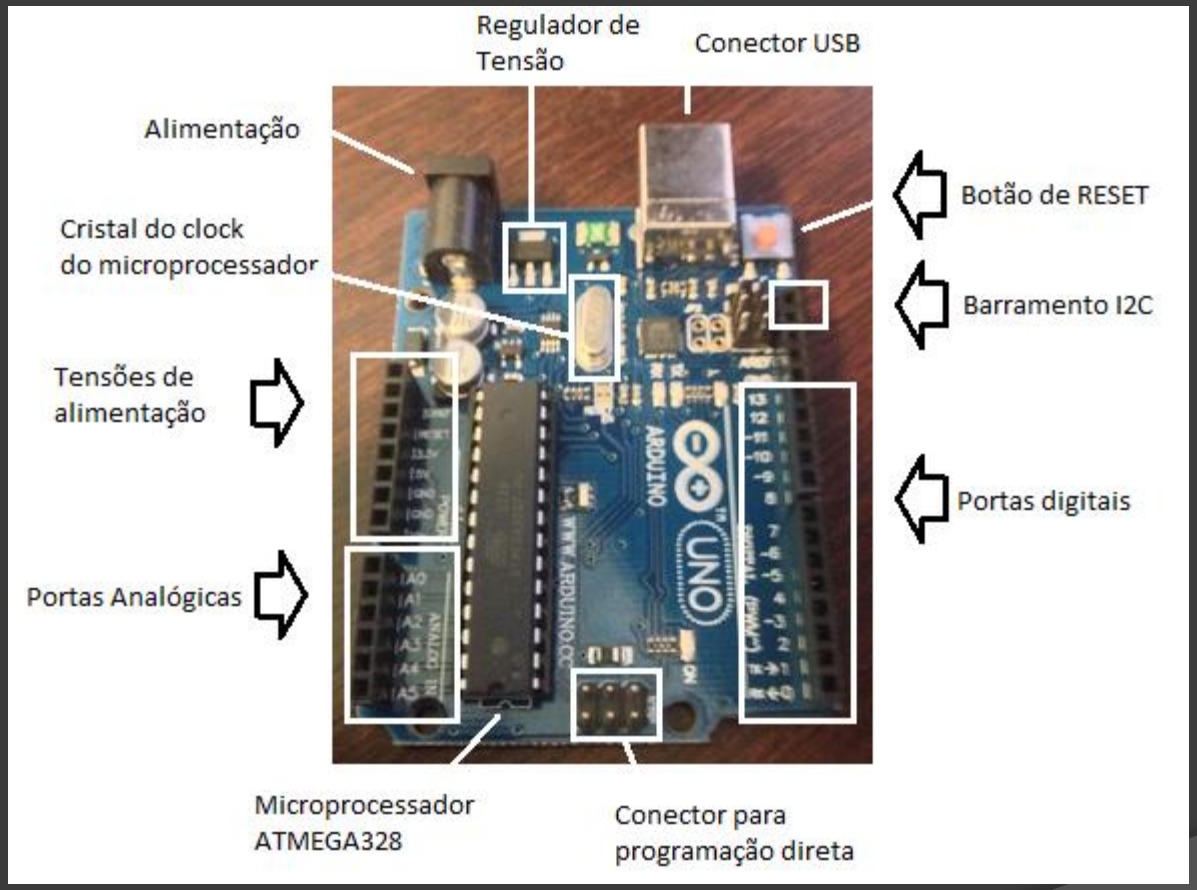
unipampa

Universidade Federal do Pampa

www.arduino.cc/

Pedro Fernando Dorneles
Edson Kakuno
Novembro de 2014

Componentes



Fonte: http://www.arduinodolito.com.br/

Características Básicas

Microcontrolador	ATmega328 / ATmega168
Pinos de entrada analógica	6
Pinos de I/O digitais	14 (dos quais 6 podem ser saídas PWM)
Tensão operacional	$5\ V$
Tensão de alimentação (recomendada)	$7 - 12\ V$
Tensão de alimentação (limites)	$6 - 20\ V$
Corrente contínua por pino de I/O	$40\ mA$
Corrente contínua para o pino 3.3 V	$50\ mA$

Pinos Digitais

Função Entrada

I – INPUT

pinoMode (led, INPUT)

0 – 1,0 V – Baixo (LOW)

3,0 – 5,0 V – Alto (HIGH)

Função Saída

O– OUTPUT

pinoMode (led, OUTPUT)

0,0 V – Baixo (LOW)

5,0 V – Alto (HIGH)

Pinos Analógicos (Somente Entrada)

Função Entrada I – INPUT

pinoMode (led, INPUT)

Para realização de medidas um conversor analógico digital A/D gera uma representação digital (valores discretos) de uma grandeza analógica (valores contínuos)

Tensões são convertidas em uma série de números binários (sinais digitais)

O conversor A/D do Arduino:

- É de 10 bits
- Recebe sinal de entrada analógica de tensão variável de 0,0 V a 5,0 V
- Pode assumir os valores binários de 0 (0000000000) a 1023 (1111111111) – 2^{10} = 1024 combinações
- É capaz de capturar 1024 níveis discretos de um determinado sinal
- É sensível a tensões de aproximadamente 5,0 mV (5,0 V/1023 = 4,89 mV) para tensão de referência igual a 5,0 V.
- É sensível a tensões de aproximadamente 1,1 mV (1.1 V/1023) para tensão de referência igual a 1,1 V.

Representação Decimal/Binária

0	0	0	0	0	1	0	1	0	1
2^9	2^8	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
512	256	128	64	32	16	8	4	2	1
					16		4		1

$$1023 = 2^9 + 2^8 + 2^7 + 2^6 + 2^5 + 2^4 + 2^1 + 2^0$$

21 – Representação decimal da sequência binária 0000010101.

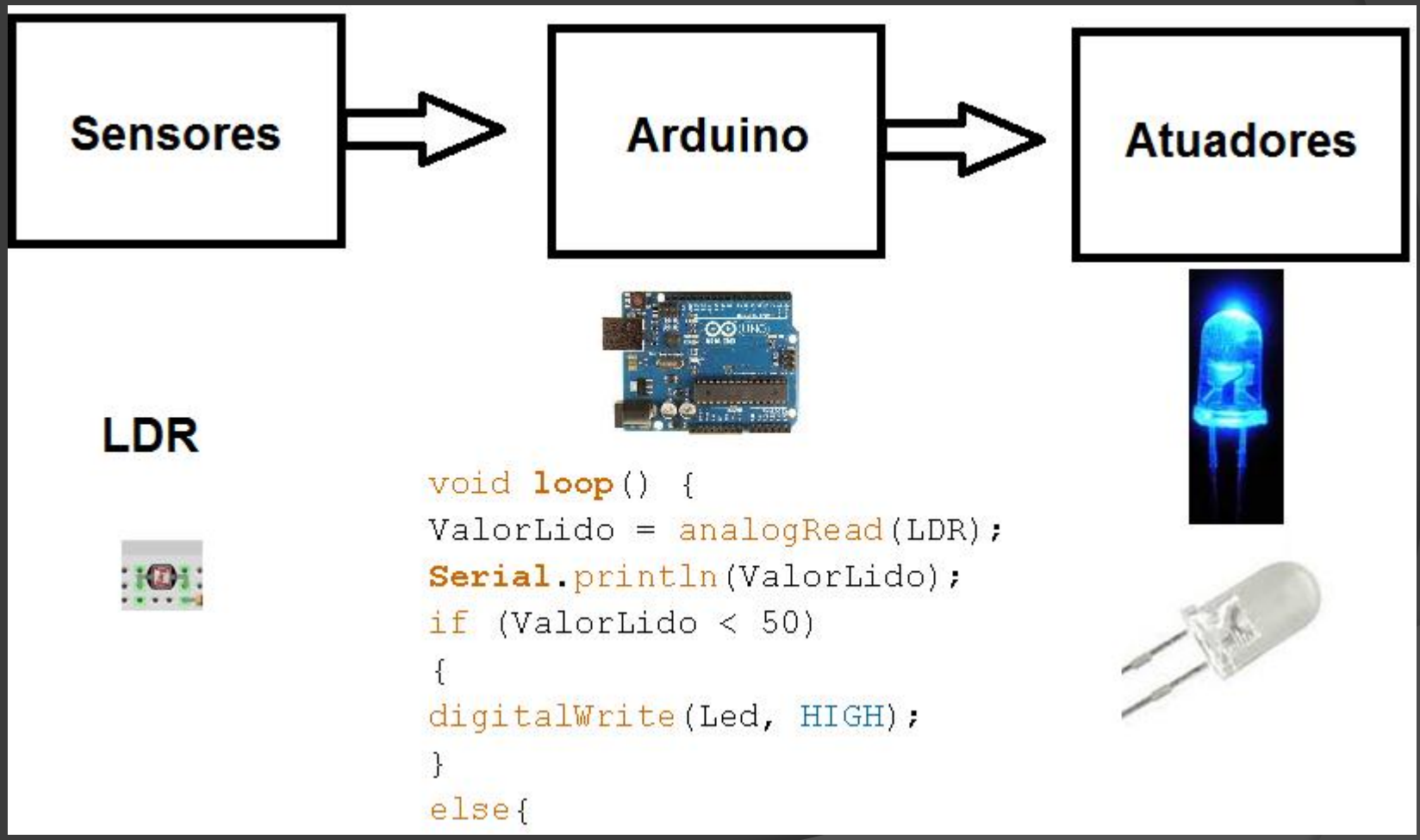
5.0 V 1023

V ? 21

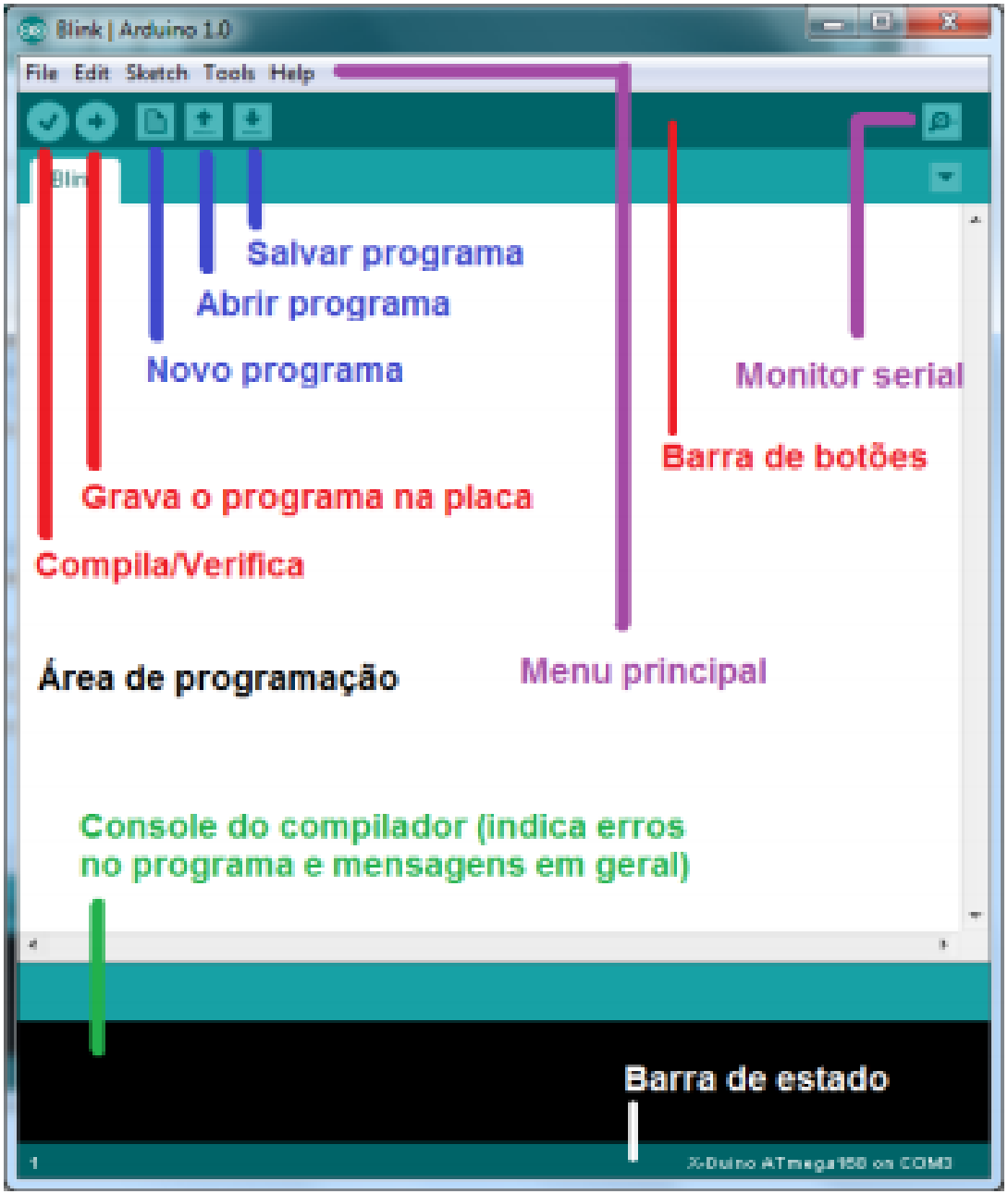
V= 5.0 21/1023 = 0,103 V = 103 mV

103/4,89 = 21,06 (vinte e um passos de 4,89 mV no ADC)

Diagrama de Blocos



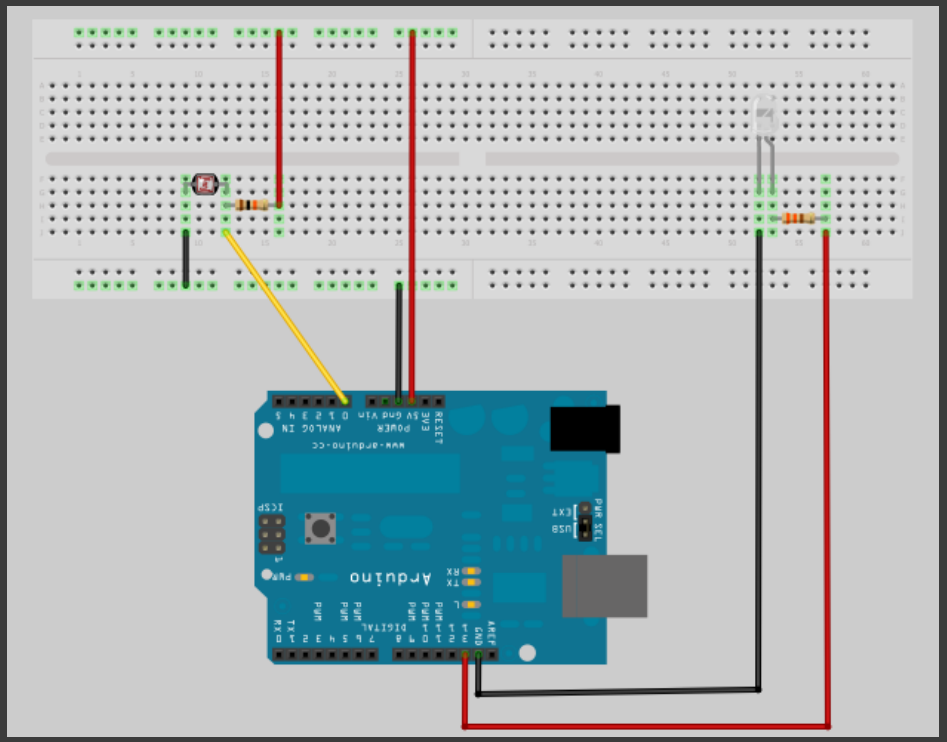
Ambiente de programação



Exemplo de programação

```
const int LDR = 0;
const int Led = 6;
int ValorLido = 0;
void setup() {
Serial.begin (9600);
pinMode(Led, OUTPUT);
}
void loop() {
ValorLido = analogRead(LDR);
Serial.println(ValorLido);
if (ValorLido < 50)
{
digitalWrite(Led, HIGH);
}
else{
digitalWrite(Led, LOW);
}
}
```

Atividade Prática

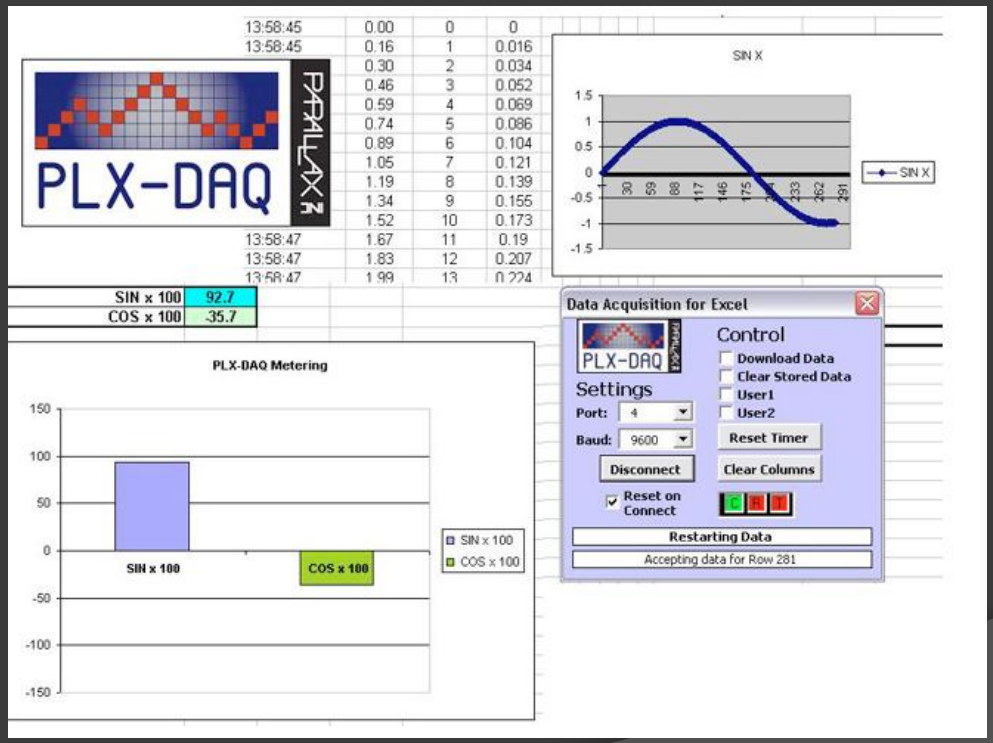


Com base no Guia do Projeto 1 monte o circuito mostrado acima, digite o código no ambiente de programação do Arduino e, finalmente, realize testes para apagar e acender o Led.

PLX-DAQ : Interface Arduino e Excel

O programa PLX-DAQ está disponível em:

http://classic.parallax.com/tabid/393/Default.aspx



Código para medida de temperatura

```
float Temp=0;
float ValorSensor=0;
float tempo=0;

void setup(){
  Serial.begin(28800);
  Serial.println("CLEARDATA");
  Serial.println("LABEL,Time,tempo,Temp");
}

void loop(){
 analogReference(INTERNAL);
 ValorSensor = analogRead(1);
 Temp = (1.1* ValorSensor * 100)/1023;
 tempo = millis();
 tempo = tempo/1000;
Serial.print("DATA,TIME,"); Serial.print(tempo); Serial.print(",");Serial.println(Temp); Serial.print(",");
 Serial.println("ROW,SET,2");
 delay(1000);
 }
```

Código para medidas de médias de temperatura

```
float Temp11=0 // variáveis
float Temp12=0;
float Temp13=0;
float Temp14=0;
float Temp15=0;
float Temp16=0;
float Temp17=0;
float Temp18=0;
float Temp19=0;
float Temp110=0;
float Temp1=0;
float tempo=0;
float ValorSensor1 = 0;
```

Setup

```
void setup(){
  Serial.begin(28800);
  Serial.println("CLEARDATA");
  Serial.println("LABEL,Time,tempo,Temp1");
}
```

Loop

```
void loop(){
  analogReference(INTERNAL);// tensão de referência (1.1 V)

  ValorSensor1 = analogRead(1);// primeira medida
  Temp11 = (1.1* ValorSensor1 * 100)/1023; // Medida em graus Celsius  - Leitura de
      1023 equivale a 1.1 V e cada 10 mV 1 ºC.
  delay(10);

  ValorSensor1 = analogRead(1);// segunda medida
  Temp12 = (1.1* ValorSensor1 * 100)/1023;
  delay(10);

  ValorSensor1 = analogRead(1);
  Temp13 = (1.1* ValorSensor1 * 100)/1023;
  delay(10);
  ...
```

Continuando

```
Temp1=(Temp11+Temp12+Temp13+Temp14+Temp15+Temp16+Temp17+
Temp18+Temp19+Temp110)/10;// média das 10 medidas
tempo = millis();
 tempo = tempo/1000;

Serial.print("DATA,TIME,"); Serial.print(tempo); Serial.print(",");
Serial.println(Temp1); Serial.print(",");
Serial.println("ROW,SET,2");
delay(1000);
}
```

Referências

ARDUINO. Disponível em: http://www.arduino.cc/. Acesso em 10 de outubro de 2013.

CAVALCANTE, M. A., TAVOLARO, C. R. C & ELIO MOLISANI, E. Física com Arduino para iniciantes. Revista Brasileira de Ensino de Física, v. 33, n. 4. 2010.

ROCHA, F. S. & GUADAGNINI, P. H. Projeto de um sensor de pressão manométrica para ensino de física em tempo real. Trabalho submetido para publicação na Revista Brasileira de Ensino de Física.

WRASSE, A., SANTOS, R., TONEL, A. P., KAKUNO, E. M. & DORNELES, P. Carrinho automatizado como recurso facilitador na construção e interpretação de gráficos da cinemática. In: XX SIMPÓSIO NACIONAL DE ENSINO DE FÍSICA – SNEF 2013 – São Paulo, SP.